Investigating Unimodal Isolated Signer-Independent Sign Language Recognition

Submitted in fulfilment

of the requirements for the degree of

Master of Science

of Rhodes University

Marc Jason Marais

Grahamstown, South Africa

March 5, 2024

Declaration of Authorship

I, Marc MARAIS, declare that this thesis titled, "Investigating Unimodal Isolated Signer-Independent Sign Language Recognition" is my own work. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: 05 March 2024

Abstract

Sign language serves as the mode of communication for the Deaf and Hard of Hearing community, embodying a rich linguistic and cultural heritage. Recent Sign Language Recognition (SLR) system developments aim to facilitate seamless communication between the Deaf community and the broader society. However, most existing systems are limited by signer-dependent models, hindering their adaptability to diverse signing styles and signers, thus impeding their practical implementation in real-world scenarios.

This research explores various unimodal approaches, both pose-based and vision-based, for isolated signer-independent SLR using RGB video input on the LSA64 and AUTSL datasets. The unimodal RGB-only input strategy provides a realistic SLR setting where alternative data sources are either unavailable or necessitate specialised equipment. Through systematic testing scenarios, isolated signer-independent SLR experiments are conducted on both datasets, primarily focusing on AUTSL – a signer-independent dataset.

The vision-based R(2+1)D-18 model emerged as the top performer, achieving 90.64% accuracy on the unseen AUTSL dataset test split, closely followed by the pose-based Spatio-Temporal Graph Convolutional Network (ST-GCN) model with an accuracy of 89.95%. Furthermore, these models achieved comparable accuracies at a significantly lower computational demand. Notably, the pose-based approach demonstrates robust generalisation to substantial background and signer variation. Moreover, the pose-based approach demands significantly less computational power and training time than vision-based approaches. The proposed unimodal pose-based and vision-based systems were concluded to both be effective at classifying sign classes in the LSA64 and AUTSL datasets.

ACM Computing Classification System

Thesis classification under the ACM Computing Classification System[1] (2012 version valid through 2024):

- **Computing methodologies** ∼ **Neural networks**
- **Computing methodologies** ∼ **Object identification**
- **Computing methodologies** ∼ **Activity recognition and understanding**

General-Terms: Convolutional Neural Networks, Video Vision Transformers, Graph Convolutional Networks, Action Recognition, Sign Language Recognition.

[1] https://www.acm.org/publications/class-2012

Acknowledgements

The authors would like to thank the authors of the publicly available datasets used in this paper. This work was undertaken in the Distributed Multimedia CoE at Rhodes University, with financial support from Telkom SA. The authors acknowledge that opinions, findings and conclusions or recommendations expressed here are those of the author(s) and that none of the above mentioned sponsors accept liability whatsoever in this regard.

I would also like to acknowledge all those who directly and indirectly supported me through my work. In particular, I would like to thank my supervisor Prof. Dane Brown, for his encouragement, guidance and support throughout the project, and my co-supervisor Mr. James Connan, for his invaluable input and experience in the sign language domain. I would also like to thank my friends and family for their support and guidance. Finally, I would like to thank my wife for her continuous emotional support and patience with me throughout this project and journey.

Contents

1 Introduction **1**

1.1 Context of Research 1

1.2 Motivation for this Research 3

1.3 Problem Statement 4

1.4 Research Question 4

1.5 Research Objectives 5

1.6 Scope and Limits 5

1.7 Assumptions 6

1.8 Limitations 6

1.9 Thesis Outline 7

2 Concepts and Literature Review **8**

2.1 Sign Language Recognition 8

2.1.1 Sign Language Syntax 8

2.1.2 Signer-Independence 10

2.2 Related Studies 11

2.2.1 Isolated Sign Language Recognition 11

2.2.2 Isolated Signer-Independent Sign Language Recognition 17

2.2.3 Critical Analysis of the Related Studies 21

2.3 Sign Language Recognition Machine Learning Architectures and Algorithms 23

2.3.1 Convolutional Neural Networks 24

2.3.1.1 ResNet 24

2.3.1.2 Inception 25

2.3.1.3 Spatiotemporal Convolutions 27

2.3.2 Pose Estimation 29

2.3.2.1 MediaPipe Hands 29

2.3.2.2 MediaPipe Pose 30

2.3.3 Graph Neural Networks 31

2.3.4 Recurrent Neural Networks 33

2.3.4.1 Long Short-Term Memory 34

2.3.4.2 Gated Recurrent Unit 35

2.3.5 Vision Transformers 36

2.3.5.1 Video Vision Transformer 37

2.3.5.2 Video Swin Transformer 38

2.3.6 Model Regularisation 39

2.3.6.1 Data Augmentation 39

2.3.6.2 Dropout 40

2.3.6.3 Batch Normalisation 41

2.3.6.4 Weight Decay 42
2.3.6.5 Early Stopping 42
2.4 Summary 43

3 Methodology and Implementation 44

3.1 Methodology 44
3.1.1 Methodology Overview 44
3.1.2 Architecture Selection 45
3.1.2.1 Pose Estimation 45
3.1.2.2 Graph Convolutional Networks 46
3.1.2.3 CNN Architectures 46
3.1.2.4 Vision Transformers 47
3.1.3 Dataset Selection 47
3.2 Implementation 48
3.2.1 Architectures 49
3.2.2 Pose-based Approach 49
3.2.2.1 Generate Pose Landmarks 50
3.2.2.2 Landmark Post-processing 50
3.2.2.3 Adjacency Matrix 51
3.2.2.4 ST-GCN 52
3.2.3 Vision-based Approaches 53

3.2.3.1 ResNet50-LSTM . . . 53
3.2.3.2 InceptionV3-LSTM . . . 54
3.2.3.3 R(2+1)D-18 . . . 55
3.2.3.4 Swin3D-T . . . 55
3.2.4 Video Dataset Sampling . . . 56
3.2.5 Model Regularisation . . . 58
3.2.5.1 Data Augmentation . . . 58
3.2.5.2 Early Stopping . . . 59
3.2.6 Parameter Tuning . . . 59
3.2.7 Training and Evaluation . . . 60
3.2.7.1 Training . . . 60
3.2.7.2 Evaluation . . . 61
3.3 Summary . . . 61

4 Experimental Setup 62

4.1 Evaluating Classification Performance . . . 62
4.1.1 Accuracy . . . 63
4.1.2 Precision and Recall . . . 63
4.1.3 F-score . . . 64
4.2 Datasets . . . 65
4.2.1 LSA64 . . . 65

4.2.2 AUTSL 66
4.3 System Infrastructure 68
4.3.1 Hardware 68
4.3.2 Software 69
4.4 Overview of Experiments 69
4.4.1 Preliminary Experiments 69
4.4.2 Evaluation of the Pose-based Approach 70
4.4.3 Evaluation of Vision-based Approaches 70
4.4.4 Evaluation of Transfer Learning 71
4.4.5 Number of Video Frames Selected 71
4.5 Summary 72

5 Results and Discussion **73**

5.1 Preliminary Experiments 73
5.1.1 Experiment 1.1 – Feature Extraction Methods 73
5.1.2 Experiment 1.2 – ST-GCN Temporal Kernel Size 74
5.1.3 Experiment 1.3 – Data Augmentation 75
5.1.4 Discussion of Experiment 1 76
5.2 Evaluation of the Pose-based Approach 77
5.2.1 Experiment 2.1 – ST-GCN on LSA64 78
5.2.2 Experiment 2.2 – ST-GCN on AUTSL 79

5.2.3 Experiment 2.3 – ST-GCN Frame Interpolation 80
5.2.4 Discussion of Experiment 2 81
5.3 Evaluation of Vision-based Approaches 83
5.3.1 Experiment 3.1 – Vision-based Approaches on LSA64 83
5.3.2 Experiment 3.2 – Vision-based Approaches on AUTSL 84
5.3.3 Discussion of Experiment 3 86
5.4 Evaluation of Transfer Learning 88
5.4.1 Experiment 4.1 – Transfer Learning on LSA64 88
5.4.2 Experiment 4.2 – Transfer Learning on AUTSL 88
5.4.3 Discussion of Experiment 4 89
5.5 Number of Video Frames Selected 90
5.5.1 Experiment 5.1 – ST-GCN Optimal Frame Selection 91
5.6 Summary 91

6 Conclusion and Future Work **93**

6.1 Conclusion 93
6.2 Contributions 95
6.2.1 Research Questions 95
6.2.2 Research Objectives 96
6.3 Future Work 98

References **99**

Appendices **111**

A Code Listings **111**

A.1 ST-GCN . . . 111

A.2 ResNet50-LSTM . . . 112

A.3 InceptionV3-LSTM . . . 112

A.4 R(2+1)D-18 . . . 113

A.5 Early Stopping . . . 114

A.6 Model Training . . . 115

A.7 Model Evaluation . . . 116

B Additional Results **117**

B.1 Experiment 2.1 – ST-GCN Classification Report LSA64 . . . 117

B.2 Experiment 2.2 – ST-GCN Classification Report AUTSL . . . 118

B.3 Experiment 3.1 – R(2+1)D-18 Classification Report LSA64 . . . 122

B.4 Experiment 3.2 – R(2+1)D-18 Classification Report AUTSL . . . 123

B.5 Experiment 2 and 3 – Confusion Matrices . . . 127

C Datasets **128**

C.1 AUTSL Dataset . . . 128

C.1.1 AUTSL Backgrounds . . . 128

C.1.2 AUTSL Sign Classes . . . 128

List of Figures

2.1 Skeleton Aware Multi-modal SLR (SAM-SLR) framework visualising the multi-modal RGB and depth ensemble architecture approach (Jiang *et al.*, 2021). 18

2.2 Hand cropping utilising OpenPose wrist and elbow landmarks for more robust cropping of both hands (De Coster *et al.*, 2021). 20

2.3 The ResNet residual block showing shortcut connection x(He *et al.*, 2016). 25

2.4 Inception module showcasing the multiple parallel convolutional filters (Szegedy *et al.*, 2015). 26

2.5 Comparison of the (2+1)D convolutional block and 3D convolutional block. 28

2.6 R(2+1)D network architecture where the (2+1)D convolutions are ResNets. 28

2.7 MediaPipe Pose and Hand topologies (Bazarevsky *et al.*, 2020, Zhang *et al.*, 2020) 30

2.8 The spatiotemporal graph visualising a human skeleton sequence as input to the ST-GCN architecture (Yan *et al.*, 2018). The solid blue dots indicate the body joints with the intra-body edge connection between joints. The temporal inter-frame edges connect the same joints between consecutive frames, represented by the faded blue dots. 32

2.9 A layer of recurrent neurons 2.9a unrolled through time 2.9b. Each neuron receives both the input vector $\mathbf{x}_t$ and the output vector from the previous time step $\mathbf{y}_{t-1}$, at each time step (Géron, 2019). 33

2.10 Single LSTM memory block depicting the gates and activation functions (Ouassil *et al.*, 2022). 34

2.11 Gated Recurrent Unit visualising the update and reset gates in a single unit (Cho *et al.*, 2014). 36

2.12 An overview of the Tubelet Embedding module with dimensions $t \times h \times w$. 38

2.13 3D shifted windows within the Video Swin Transformer. 39

2.14 Visualisation of Neural Network before and after the application of dropout regularisation (Srivastava *et al.*, 2014). Dropped neurons are represented with crosses. 41

3.1 MediaPipe Holistic visualisation of landmarks. 46

3.2 High-level systematic overview of the SLR system. 49

3.3 Video clip pose landmark generation with subsequent imputation for each landmark subset. The final output is a NumPy array for each video frame. 50

3.4 Comparison of failed frame detections for the LSA64 and AUTSL datasets with MediaPipe Holistic. 51

3.5 Comparison of sample video frames with and without graph connections. The first row depicts the landmarks without any connections, whereas the second row adds the connections from the adjacency matrix. 52

3.6 ST-GCN architecture (Ghosh *et al.*, 2022). 53

3.7 Architecture of the tiny version (Swin3D-T) of Video Swin Transformer. . 56

4.1 Snapshots extracted from the LSA64 dataset. 65

4.2 AUTSL sample video snapshots. 67

4.3 Distribution of samples per signer across the entire AUTSL dataset. 68

5.1 Experiment 1.1 – Accuracy performance of the architectures on the LSA64 preliminary dataset. 74

5.2 Experiment 1.2 – Hyperparameter tuning test accuracy of temporal size for the ST-GCN model on the LSA64 and AUTSL datasets. 75

5.3 Segmentation of both hands on the Appear sign from the LSA64 dataset. . 76

5.4 Experiment 2.1 – Training and validation accuracy performance of the ST-GCN pose-based approach on the LSA64 dataset. 78

5.5 Experiment 2.1 – ST-GCN confusion between the predicted *Buy* class (top row) and actual *Realize* class (bottom row). For comparison, five sequential frames were selected at even intervals across the entire video clip from the LSA64 test split. 79

5.6 Experiment 2.2 – Training and validation accuracy performance of the ST-GCN pose-based approach on the AUTSL dataset. 80

5.7 Experiment 2.2 – ST-GCN confusion between the predicted *Atatürk* class (top row) and actual *Government* class (bottom row). For comparison, five sequential frames were selected at even intervals across the entire video clip from the AUTSL test split. 81

5.8 Experiment 3.1 – Comparison of vision-based models performance on the LSA64 dataset. 83

5.9 Experiment 3.2 – Comparison of vision-based models performance on the AUTSL dataset. 85

5.10 Experiment 3.2 – R(2+1)D-18 confusion between the predicted *Soup* class (top row) and actual *School* class (bottom row). For comparison, five sequential frames were selected at even intervals across the entire video clip from the AUTSL test split. 85

5.11 Experiment 3.2 – R(2+1)D-18 confusion between the predicted *Police* class (top row) and actual *Rent* class (bottom row). For comparison, five sequential frames were selected at even intervals across the entire video clip from the AUTSL test split. 86

5.12 Experiment 4.2 – R(2+1)D-18 model accuracy performance when trained from scratch and with transfer learning from the Kinetics-400 dataset on the AUTSL dataset 89

C.1 Examples of the 20 different background variations from AUTSL. 129

List of Tables

2.1 Pose estimator impact of imputation and normalisation. 14

2.2 Top-K accuracy for each method on the Flemish Sign Language Corpus (De Coster *et al.*, 2020). 16

3.1 Dataset folder structure. 57

4.1 Confusion Matrix Explanation. 62

4.2 Breakdown of the LSA64 signs with the corresponding hand utilised to perform the sign, ordered by ID. R refers to the right hand, and B refers to both hands. 66

4.3 AUTSL dataset statistics. 67

5.1 Experiment 1.3 – R(2+1)D-18 Model Accuracy Performance with and without Augmentations. 76

5.2 Experiment 2.3 – Comparison of the post-processing imputation of landmarks on the LSA64 and AUTSL dataset using the ST-GCN model. 81

5.3 Model accuracy performance compared to existing studies on the AUTSL test dataset using RGB data. 87

5.4 Experiment 4.1 – R(2+1)D-18 Model Accuracy Performance with and without Transfer Learning from the Kinetics-400 dataset on the LSA64 dataset. 88

5.5 Experiment 5.1 – Comparison of performance of ST-GCN model based on the number of video frames selected. 91

B.1 Experiment 2.1 – ST-GCN classification report for LSA64 dataset. 117

B.2 Experiment 2.2 – ST-GCN classification report for AUTSL dataset. 118

B.3 Experiment 3.1 – R(2+1)D-18 classification report for LSA64 dataset. . . . 122

B.4 Experiment 3.2 – R(2+1)D-18 classification report for AUTSL dataset. . . 123

C.1 AUTSL sign classes with the number of hands used to perform each sign and the number of training samples of each sign. 130

Listings

3.1 Data augmentation pipeline. 59
3.2 Training loss function and optimiser. 60
A.1 ST-GCN block. 111
A.2 ResNet50-LSTM model with the sequential feeding of the ResNet50 model output features to the LSTM mode. 112
A.3 InceptionV3-LSTM model with the sequential feeding of the InceptionV3 model output features to the LSTM model. 112
A.4 R(2+1)D BasicBlock. 113
A.5 PyTorch early stopping callback. 114
A.6 Model training pipeline. 115
A.7 Model testing and accuracy reporting. 116

Declaration of Publications

The following research papers have been published, and parts of their materials are included in this thesis:

1. **Marc Marais**, Dane Brown, James Connan and Alden Boby. An Evaluation of Hand-Based Algorithms for Sign Language Recognition. In *2022 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)*, pages 1–6. IEEE, 2022.

2. **Marc Marais**, Dane Brown, James Connan, Alden Boby and Luxolo Lethukuthula Kuhlane. Investigating Signer-Independent Sign Language Recognition on the LSA64 Dataset. In *Southern Africa Telecommunication Networks and Applications Conference (SATNAC) 2022*, pages 183–188. SATNAC, 2022.

3. **Marc Marais**, Dane Brown, James Connan and Alden Boby. Improving Signer-Independence Using Pose Estimation and Transfer Learning for Sign Language Recognition. *Advanced Computing. Communications in Computer and Information Science*, vol. 1782, 2023, Springer Cham, pp. 415–428. doi: 10.1007/978-3-031-35644-5_34

4. **Marc Marais**, Dane Brown, James Connan and Alden Boby. Spatiotemporal Convolutions and Video Vision Transformers for Signer-Independent Sign Language Recognition. In *2023 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)*, pages 1–6. IEEE, 2023.

5. **Marc Marais**, Dane Brown, James Connan and Alden Boby. Facial Liveness and Anti-Spoofing Detection using Vision Transformers. In *Southern Africa Telecommunication Networks and Applications Conference (SATNAC) 2023*, pages 187–192. SATNAC, 2023.

Glossary

ASL	American Sign Language
AUTSL	Ankara University Turkish Sign Language
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
FPS	Frames per Second
GCN	Graph Convolutional Network
GNN	Graph Neural Network
GRU	Gated Recurrent Unit
LSTM	Long short-term Memory
PTN	Pose Transformer Netowrk
RNN	Recurrent Neural Network
SASL	South African Sign Language
SLR	Sign Language Recognition
ST-GCN	Spatio-Temporal Graph Convolutional Network
ViT	Vision Transformer
ViViT	Video Vision Transformer
VTN	Video Transformer Network

1

Introduction

1.1 Context of Research

The primary means of communication for the Deaf and Hard of Hearing community are hand gestures and sign language, posing a significant challenge in communicating with them (Cooper *et al.*, 2011). Sign language is a visual-gestural communication system used by deaf individuals, employing hand movements, facial expressions, and body language to convey meaning. The Deaf and Hard of Hearing community, which consists of individuals who use sign language as their primary mode of communication, forms a rich cultural and linguistic group with a shared sense of identity and unique experiences (Shoham and Heber, 2012). In South Africa, English and the South African Sign Language (SASL) are spoken and recognised as official languages. However, it is important to note that English and SASL are completely unrelated (Stokoe, 1980).

The linguistics of sign language is as complex as any natural spoken language (Mayberry and Squires, 2006). Sign languages differ depending on each region; they do not mimic spoken languages and implement their syntax using glosses. A gloss is a label associated with each sign or a sequence of signs that conveys the sign in written or spoken representation (Rastgoo *et al.*, 2021b). Sign languages can be characterised through a topic-comment structure, where a topic or scene is set up and subsequently commented on. Signing consists of three vital elements: manual features, non-manual features, and fingerspelling (Cooper *et al.*, 2011). Manual features consist of hand gestures, utilising hand shape and motion to convey meaning. Non-manual features included facial expressions, body posture, fingerspelling, and words from the local verbal language spelt out

using gestures. These three elements are grouped as a gloss, the fundamental building block for Sign Language Recognition (SLR).

SLR has two types of models: isolated SLR models and continuous SLR models (Sharma *et al.*, 2021). Both models incorporate static and dynamic gestures, where static gestures involve stationary hand positions, while dynamic gestures involve moving the hands or body to convey information in sign language (Hirafuji Neiva and Zanchettin, 2018). Isolated SLR recognises words or letters through dynamic or static gestures (Aloysius and Geetha, 2020). In contrast, continuous SLR is the problem of recognising a sequence of sign language glosses and each sign's temporal boundaries, utilising glosses from annotated video sequences. Data collection methods for SLR consist of hardware-based and vision-based approaches. Hardware-based SLR entails using specialised equipment for collecting sensor data or necessitates the signer to wear markers or gloves. In contrast, vision-based approaches rely only on camera input without needing specialised equipment.

Signer-independent SLR refers to the ability of a computer system to recognise and interpret sign language gestures made by different signers, regardless of their individual signing style or speed (Von Agris *et al.*, 2008). This task is challenging because sign language can vary significantly from person to person and even within the same signer, depending on the context and intended message. Researchers typically use machine learning algorithms to train a system on large datasets of sign language gestures from many different signers to achieve SLR. Such systems aim to recognise the common underlying features of the gestures, e.g. hand shape, movement, and location, rather than relying on specific signing styles (Adeyanju *et al.*, 2021).

In summary, signer-independent SLR is a field of study that aims to develop computer systems that recognise sign language gestures made by different signers, regardless of their individual signing styles.

1.2 Motivation for this Research

Sign language communication requires a significant amount of teaching and effort to master. Automated SLR aims to bridge the communication gap between sign language users and others through the automated recognition of signs from videos. Hardware-based SLR often involves using expensive equipment to gather sensor data or requires the signer to wear markers, which is impractical in real-world scenarios. Therefore, vision-based SLR opens the door to markerless and easily accessible methods for SLR.

Unimodal machine learning refers to a model that utilises a single data source or type of input to learn and make predictions. On the other hand, multi-modal models utilise more than one data source to learn and make predictions. Different data sources are typically unavailable in a real-world setting, making unimodal models simpler and computationally more efficient. The single data source for SLR would be video RGB input, where RGB refers to the Red, Green, and Blue colour representation. The problem can be further broken down into pose-based and vision-based approaches from this single RGB input. Pose-based approaches incorporate pose estimation algorithms, and vision-based approaches apply no algorithms to the RGB input.

The vast growth in computer vision techniques in the past decade has made advancements in human pose estimation and transfer learning, which could lead to improvements in signer-independent SLR. Human pose estimation utilises key points on a person to infer the person's pose in an image or video (Munea *et al.*, 2020). Video RGB input in an SLR system must efficiently handle sampling key frames uniformly to find a balance between system performance and computational resources. Furthermore, the growing concept of transfer learning, which leverages knowledge gained from training a model on one task and applying it to improve performance on a different but related task, could significantly improve vision-based SLR models.

There has been significant progress in signer-dependent isolated SLR in recent years. Signer-dependent SLR systems can achieve high accuracy rates because they can learn the unique signing style of an individual signer. However, they are limited in their ability

to generalise to new signers.

In contrast, signer-independent SLR involves training a system to recognise signs from multiple signers without requiring specific training for each signer. Signer-independent SLR is a more challenging problem because it requires the system to learn the variability of signs across different signers. Furthermore, achieving comparable accuracy in unimodal signer-independent SLR, as opposed to multi-modal, is inherently more challenging. However, unimodal approaches are significantly more computationally expensive. Therefore, it has the potential to be more useful in practical applications (Von Agris *et al.*, 2008).

1.3 Problem Statement

This research investigates the applicability of combining pose estimation algorithms with deep learning techniques for signer-independent isolated SLR. Furthermore, it compares the effectiveness of unimodal pose-based and vision-based methods for SLR in successfully detecting isolated sign language from video input.

The hypothesis is that a system can be constructed to recognise signer-independent isolated sign language utilising pose estimation to improve accuracy and computational efficiency on RGB videos.

1.4 Research Question

The overarching research question is thus: 'Which methods are effective for optimising signer-independent SLR in RGB videos?' This can be broken into specific sub-questions:

1. Do pose-based algorithms improve signer-independent SLR performance over vision-based algorithms?

2. Are pose-based algorithms more computationally efficient than vision-based algorithms?

3. Does applying transfer learning improve signer-independent SLR model performance?

4. How does the number of video frames uniformly selected from each video clip affect model performance?

1.5 Research Objectives

This research aims to achieve the following objectives:

1. Progressively collect suitable isolated signer-independent SLR datasets and determine the most effective way to build an SLR system.

2. Design and construct a relatively inexpensive system, in terms of computational resources, that utilises real-world data and performs isolated signer-independent SLR.

3. Compare the different feature extraction methods for SLR.

4. Measure the effectiveness of the frame interpolation for pose-based signer-independent SLR.

5. Determine the effectiveness of different vision-based approaches for signer-independent SLR.

6. Observe the effect of applying transfer learning to the vision-based models.

7. Evaluate the effectiveness of unimodal pose-based and vision-based approaches for signer-independent SLR.

1.6 Scope and Limits

The summarised approach taken for this thesis is as follows. Initially, various literature on machine and deep learning algorithms, pose estimation algorithms, and datasets for

SLR will be analysed. Subsequently, the best theoretical approaches will be chosen for effective isolated signer-independent SLR and their potential for enhancement with pose estimation algorithms. A combination of the reviewed methods in the literature with pose estimation algorithms will be utilised to design an effective vision-based SLR system.

To measure the success of the proposed system, the appropriate machine learning metrics applicable to SLR will be chosen for evaluation and fair comparison with existing systems. Optimal model parameters undergo systematic tuning dependent on each experiment. Various experiments will be conducted on the selected datasets to compare the effectiveness and generalisation ability of the proposed algorithms for isolated signer-independent SLR.

1.7 Assumptions

The experiments are conducted under the following assumptions:

- Sign language videos in datasets used in training and testing will contain the entire upper body of the signer from either a face-on or side view.
- As this study focuses on unimodal SLR using only RGB video data, signers are assumed not to use special equipment such as sensors, data gloves, or coloured markers. It should be emphasised that achieving comparable accuracy in unimodal SLR, as opposed to multi-modal SLR, is inherently more challenging, and this distinction underscores the significance and complexity of the undertaken research.

1.8 Limitations

Processing any natural language, including sign language, is a complex task due to the many rules and relationships that are difficult to formulate mathematically to program into computers. Hence, SLR fares well with signer-dependent single character and word recognition but may struggle with more complex signer-independent SLR.

1.9 Thesis Outline

The remainder of this thesis is arranged as follows:

Chapter 2: *Concepts and Literature Review*: A broad overview of signer-independent SLR is given with related studies on isolated SLR and isolated signer-independent SLR. The machine learning concepts and algorithms utilised in this thesis and related studies are also explained.

Chapter 3: *Methodology and Implementation*: This chapter presents the proposed approaches towards unimodal signer-independent SLR. The implementation of each architecture and training pipeline are subsequently discussed.

Chapter 4: *Experimental Setup*: This chapter details the experimental setup used in this research. The appropriate metric selection is initially discussed, followed by the SLR dataset selection used in the experiments, and finalised with the design of the various experiments conducted.

Chapters 5: *Results and Discussion*: Preliminary experiments and validation of machine learning architecture parameters are presented, followed by the results achieved by the proposed approaches, along with an analysis and discussion.

Chapter 6: *Conclusion*: This chapter concludes the thesis, emphasising the contributions made towards the research in SLR and outlining potential avenues for future work.

2

Concepts and Literature Review

This chapter gives a broad overview of SLR and signer-independent isolated SLR. The relevant concepts and techniques from related studies are detailed to elaborate on the knowledge required for implementing signer-independent SLR. Section 2.1 details the syntax behind SLR and highlights the important aspects of signer-independent SLR. The related studies in SLR and signer-independent are discussed in Section 2.2 and Section 2.3 presents machine learning algorithms and architectures relevant to SLR.

2.1 Sign Language Recognition

Communication between the Deaf and Hearing is dependent on using sign language, incorporating a range of hand and body gestures. The linguistics behind sign language is as complex as any natural spoken language. Therefore, communication between the Hearing and the Deaf communities requires mastering sign language. Consequently, SLR aims to alleviate the need to master sign language and bridge the communication gap.

2.1.1 Sign Language Syntax

Sign language syntax refers to the rules and conventions that govern the order and structure of signs used in sign languages. These rules differ from those that apply to spoken languages, as sign languages use a visual-spatial medium to convey meaning. Fundamentally, sign languages can be characterised through a topic-comment structure, where a topic or scene is set up and subsequently commented on (Cooper *et al.*, 2011).

Some key aspects of sign language syntax include (Rastgoo *et al.*, 2021a):

- Word order: Sign languages often have a flexible word order, allowing speakers to convey meaning in different ways. However, general rules still govern the order of signs in a sentence. For example, in American Sign Language (ASL), the subject often comes before the object, and the verb comes at the end of the sentence. The subject-object-verb sign order is also exhibited in the Argentine and Turkish sign languages (Massone and Curiel, 2004).

- Manual markers: Features consist of hand gestures, utilising hand shape and motion to convey meaning.

- Facial expressions: Facial expressions are important to sign language syntax. They can convey emotions, clarify meaning, and indicate grammatical features such as tense and aspect.

- Non-manual markers: In addition to facial expressions, sign languages use a variety of non-manual markers, such as body position, eye gaze, and head movements, to convey meaning and indicate grammatical features.

- Visual-Spatial Representation: Visual-Spatial representation covers the size, shape, or movement of an object or action performed by a signer. Subsequently, providing additional information about the object or action being referred to.

- Sign Gloss: A label associated with each sign or a sequence of signs formed by grouping manual and non-manual features (Rastgoo *et al.*, 2021b).

Sign language syntax is complex and varies across different sign languages. However, the underlying principles are based on the visual-spatial nature of sign languages, incorporating the aspects given above and the need to convey meaning in a way accessible to the Deaf. Moreover, signs performed by signers with different competency and proficiency levels have the same underlying structure with slight variation, which needs to be identified by a robust SLR system.

2.1.2 Signer-Independence

Von Agris *et al.* (2006, 2008) identified signer-independent SLR as the process of recognising signs performed by different signers without requiring specific training for each signer. Whereas signer-dependent SLR involves training a system to recognise the signs of a specific signer.

They established the following key differences between signer-independent and signer-dependent SLR after a comprehensive survey:

- Training data: Signer-independent SLR systems require training data from multiple signers. Signer-dependent systems tend to be more accurate for the specific signer they are trained on. However, due to their dependent nature, they may not generalise well to new signers.
- Variability: Signer-independent SLR systems must account for the variability of signs across different signers, making the problem more challenging. The variability may include:
  - fluent versus native signers
  - signing speed
  - hand shape
  - co-articulation between signs
- Practical applications: Signer-independent SLR systems have the potential to be more beneficial for practical applications, as they can be applied to a broader range of signers without the need for individualised training.

There have been some studies on signer-independent SLR, but they are relatively few compared to those on signer-dependent SLR. One reason is that collecting datasets of signers from different backgrounds for each language is difficult and time-consuming. Additionally, the variability of sign language across different signers makes it challenging to develop robust and accurate signer-independent SLR systems.

Overall, both signer-independent and signer-dependent SLRs have their strengths and weaknesses, and the choice of approach will depend on the specific application and context in which the technology is being used. However, as SLR technology advances, there is increasing interest in developing more robust and accurate signer-independent SLR systems that can be used for a wider range of signers and applications.

2.2 Related Studies

Al-qurishi *et al.* (2021) surveyed deep learning trends by analysing SLR studies from 2014–2021, covering aspects from data collection, current computer vision techniques, benchmarks and open issues for both machine learning and deep learning techniques.

Traditional machine learning methods predominately combine feature extraction techniques such as Principal Component Analysis, Linear Discriminant Analysis, or Histogram of Oriented Gradients combined with a classification algorithm, namely support vector machines. However, traditional machine learning approaches neglect temporal information, thus affecting video-based systems such as SLR that require a spatial and temporal component.

Therefore, there has been a shift towards deep learning algorithms that include spatial and temporal information. Commonly implemented deep learning algorithms include convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

2.2.1 Isolated Sign Language Recognition

Isolated SLR, also referred to as word-level SLR, is the recognition of words or letters through dynamic or static gestures (Aloysius and Geetha, 2020). In the pursuit of enhancing communication between the Deaf and Hearing community, isolated SLR is a pivotal component in the development of technologies geared towards recognising and interpreting isolated sign expressions.

Konstantinidis *et al.* (2018) implemented a system for pose-based SLR using RGB input by separating the extraction of body skeletal landmarks and hand skeletal landmarks. Hand skeletal detection was achieved using a pretrained ImageNet VGG-19 neural network (Simonyan and Zisserman, 2014). Only the first ten layers of the VGG-19 network are used for body skeletal detection to avoid overfitting from overly complex modelling. Joint-line distances are calculated using the hand and body coordinates to create spatial features. The distances model each joint to its projections on the lines formed by every other skeleton joint pair. A four-stream deep neural network incorporating a stacked Long short-term memory (LSTM[1]) is employed alongside four streams of linear dynamic system histograms to extract temporal features. The LSTM and linear dynamic system descriptors comprise eight streams processed with fully connected and softmax layers for final classification. The streams of the LSTMs and linear dynamic systems are averaged to determine the class of the sign. The system achieved a 98.09% accuracy on the LSA64 dataset, an Argentinian signer-dependent sign language dataset consisting of 64 isolated signs, outperforming Ronchetti *et al.* (2016), which implemented a one versus all multi-class support vector machine and one versus all Hidden Markov Model[2], by 3.01% and 2.17%, respectively.

A study by Moryossef *et al.* (2021) compared OpenPose Single-Network Whole-Body pose estimation to MediaPipe Holistic pose estimation frameworks on the Ankara University Turkish Sign Language (AUTSL) isolated SLR dataset (Sincan and Keles, 2020). The OpenPose model utilised an SLR transformer based on Camgoz *et al.* (2020)'s work. In contrast, the MediaPipe system implemented a two-layer Bidirectional Long Short-Term Memory (BiLSTM[3]) using angles, length and raw landmarks as feature extractors. The results found that both systems achieved accuracies between 80% and 85%, with MediaPipe Holistic pose estimation framework achieving the higher accuracy. Both pose estimation frameworks struggled when presented with signs where the hands occlude each

[1]Long short-term memory forms part of RNNs using feedback connections, enabling this deep learning architecture to process entire data sequences such as video (Hochreiter and Schmidhuber, 1997).

[2]A statistical model that represents a sequence of observable events, assuming an underlying, unobservable sequence of hidden states. It is widely used in traditional machine learning for modelling temporal and sequential data (Mor *et al.*, 2021)

[3]The BiLSTM model consists of two LSTMs taking in input and running in opposite directions along the time domain (Schuster and Paliwal, 1997).

other or the face. They found that a considerable amount of information is lost through skeletal representation that has to be represented by the image domain.

De Coster *et al.* (2023) investigated different pose estimators frameworks as robust feature extractors for processing sign language videos. Three popular pose estimator frameworks were compared for SLR: OpenPose, MMPose and MediaPipe (Cao *et al.*, 2019, Sengupta *et al.*, 2020, Lugaresi *et al.*, 2019). Each framework's upper body and hand landmarks are considered while removing the face mesh from MediaPipe. Post-processing is applied to the landmarks in the form of missing landmark imputation and normalisation. The imputation consists of linear interpolation, extrapolation, and zero-based imputation. For MediaPipe, this is done separately for the right hand, left hand and the remaining pose landmarks. Normalisation accounts for differences in translation and scales the landmarks, also applied separately to each subset. The hand landmarks are centred around the wrist and scaled based on the Euclidean distance between the wrist and the knuckle of the middle finger. The remaining pose landmarks are centred on the chest and rescaled using the Euclidean distance between the shoulders. Notably, the MediaPipe holistic framework either returns estimations for the entire subset or returns none at all, whereas OpenPose individual landmarks may be missing from each subset. Notably, comparing the three pose estimation algorithms, MediaPipe processes 4.8 frames per second (FPS), OpenPose 1.1 FPS and MMPose 1.3 FPS. OpenPose and MMPose leverage a GPU, whereas MediaPipe runs on the CPU.

They compare the performance of pose-based and vision-based approaches. For the latter, a ResNet-34 feature extractor combined with a self-attention network is implemented. The findings showed that landmark post-processing significantly improved the results and enabled transfer learning across different sign languages, as visualised in Table 2.1.

The better performance of MediaPipe is primarily attributed to the holistic architecture having a dedicated hand pose estimation model (Zhang *et al.*, 2020) alongside the full-body model, compared to OpenPose and MMPose, which only utilise full-body models. Therefore, MediaPipe is typically better at landmark detection for the hands.

Most isolated SLR datasets are limited to a few words or letters. Hence, Li *et al.* (2020)

Table 2.1: Pose estimator impact of imputation and normalisation.

Pose estimator	Normalisation	Imputation	Accuracy(%)
MMPose	✗	N/A	21.47
MMPose	✓	N/A	36.49
OpenPose	✗	✗	35.53
OpenPose	✓	✗	39.27
OpenPose	✓	✓	39.23
MediaPipe	✗	✗	40.01
MediaPipe	✓	✗	47.08
MediaPipe	✓	✓	48.15

introduced the publicly available Word-Level American Sign Language (WLASL) video dataset of over 2000 words performed by over 100 different signers. They proposed two deep learning-based approaches to solving the isolated SLR problem on the WLASL dataset: vision-based and pose-based approaches.

The vision-based approach consists of two baseline networks: a VGG-GRU which is a combination of a 2D-CNN and an RNN, consisting of a stacked Gated Recurrent Unit (GRU[4]) and an Inflated 3D ConvNet (I3D[5]). The pose-based networks both utilise OpenPose to extract body landmarks, which are fed into an RNN using a stacked GRU, Pose-GRU, and a novel Pose Temporal Graph Convolution Networks (TGCN). Traditional human pose estimation utilises 2D joint angles to model motions. However, Li *et al.* (2020) proposed TGCNs, which employ a holistic representation of the trajectories of the landmarks to encode temporal motion information. A fully connected graph is used to represent a human body with K vertices and edges represented as weighted adjacency matrix $\mathbf{A} \in \mathbb{R}^{K \times K}$. The fully connected graph enables learning dependencies among joints. A graph convolution layer is expressed according to Equation 2.1.

$$\mathbf{H}_{n+1} = \mathcal{G}_n\left(\mathbf{H}_n\right) = \sigma\left(\mathbf{A}_n \mathbf{H}_n \mathbf{W}_n\right) \tag{2.1}$$

Here, $\mathcal{G}_n$ accepts matrix $\mathbf{H}_n \in \mathbb{R}^{K \times F}$ as input features representing the node features

[4]A gating mechanism in RNNs, similar to LSTMs modulating the flow of information with gating units, however without having separate memory cells (Chung *et al.*, 2014).

[5]Expanding the 2D ConvNet into 3D to enhance the learning of spatiotemporal feature extractors (Carreira and Zisserman, 2017).

or representations at the n^{th} layer in a GCN, where F represents the feature dimension output by the previous layer. The hyperbolic tangent activation function is represented by σ, where $\mathbf{A}_n$ is the adjacency matrix and $\mathbf{W}_n$ denotes a set of trainable weights. The TGCN network utilises multiple stacked residual graph convolutional blocks and averaging pooling along the temporal dimensions to represent pose trajectory features.

Mean scores of top-K[6] classification with $K \in \{1, 3, 5, 10\}$ were use to evaluate the models. Pose-GRU, Pose-TGCN and I3D achieved top-10 accuracy ranging from 70.15% to 89.92% on the smaller subsets of WLASL100, WLASL300 and WLASL1000. However, when presented with the larger WLASL2000 subset, the best-performing model was I3D with a top-10 accuracy of 66.31%, followed by Pose-TGCN with 62.24%.

De Coster *et al.* (2020) explored four different methods utilising pose estimation and transformer networks. Each method employs one or more feature extractors combined with a classifier. The training and testing were conducted using the Flemish Sign Language Corpus. The feature extractors in each method are either OpenPose (Cao *et al.*, 2019) or a 2D-CNN pretrained on the ImageNet dataset (Deng *et al.*, 2009). The four methods can be summarised as follows:

1. PoseLSTM: OpenPose exacts 137 landmarks for each video frame. This comprised 21 landmarks per hand, 70 facial landmarks, and 25 body pose landmarks. Subsequently, these landmarks are fed into an LSTM classifier for final classification.

2. Pose Transformer Network (PTN): The OpenPose landmarks are utilised as input for a multi-head attention classifier.

3. Video Transformer Network (VTN): The network is based on a 2D-CNN feature extractor combined with a multi-head attention network proposed by Kozlov *et al.* (2020). The ResNet-34 feature extractor transforms each frame into a 512-dimensional vector. The multi-head attention network classifies the resulting feature vector.

[6]The frequency where the ground truth label is in the top n of the network predictions. This offers insight into performance beyond the single most probable prediction.

Table 2.2: Top-K accuracy for each method on the Flemish Sign Language Corpus (De Coster *et al.*, 2020).

Method	**Top-1(%)**	**Top-3(%)**	**Top-5(%)**	**Top-10(%)**
PoseLSTM	54.55	66.61	72.75	79.81
PTN	61.73	75.77	81.12	87.63
VTN	73.39	85.25	89.19	92.62
MTN	74.70	86.11	89.81	93.37

4. Multimodal Transformer Network (MTN): The OpenPose and ResNet-34 features are fused using scaled dot-product attention to learn the important features and classified using the multi-head attention network.

The methods were evaluated using top-K accuracy, with $K \in \{1, 3, 5, 10\}$. The accuracy for each model is shown in Table 2.2, where transformer networks are significantly better at classifying pose landmarks than LSTM networks. The minimal difference between the VTN and MTN networks suggests that 2D CNNs can already learn salient features. However, using OpenPose landmarks in the MTN network introduces additional information to assist with class discrimination.

Hrúz *et al.* (2022) investigated isolated SLR on the AUTSL and WLASL3000 datasets, focusing on vision-based and posed-based approaches. For the vision-based approaches, they employ the Resnet50-I3D and TimeSformer[7]. SPOTER (sign pose-based transformer) is utilised for the pose-based approach (Boháček and Hrúz, 2022) built on the original Transformer architecture with modifications to work with body pose sequences. A linguistics-aware normalisation procedure is applied to the body pose estimation with added positional encoding information. OpenPose and MMPose pose estimators were considered to extract skeletal data for the SPOTER architecture. Both vision-based approaches were pretrained on the Kinetics-400 dataset (Kay *et al.*, 2017).

Results on the AUTSL test dataset found that the ResNet50-I3D architecture achieved the best accuracy of 92.86%, followed by the TimeSformer with 85.89%. The SPOTER architecture yielded 84.90% accuracy using the MMPose pose estimator, significantly better

[7] A transformer-based architecture designed explicitly for sequential data processing, integrating time information into its self-attention mechanism to capture temporal dependencies effectively. (Bertasius *et al.*, 2021)

than the 78.89% when using OpenPose. Hrúz *et al.* also considered various ensemble[8] approaches, with a novel neural ensembler[9] yielding the top accuracies – 96.37% and 70.12% on the AUTSL and WLASL3000 datasets, respectively. Using transfer learning from the AUTSL dataset and training on the WLASL3000 dataset, the TimeSformer accuracy further improved by 15.76% from 36.79 to 52.55%. However, when transfer learning from the WLASL3000 data set and training on the AUTSL dataset no significant improvement was noted. This is largely due to the vast amount of signer and background variation present in the AUTSL dataset.

Important aspects of the isolated SLR studies are that the majority of studies employ either a vision-based or pose-based approach. Where a vision-based system is implemented, architectures include hybrid CNN-RNN, 3DCNNs, or vision transformers. Pose-based approaches employ a pose estimator algorithm combined with either a TGCN or transformer network to model the relationships between landmarks in both the spatial and temporal domains correctly. Moreover, De Coster *et al.* highlighted the importance of dealing with missing frame landmarks and post-processing of landmarks for better model performance.

2.2.2 Isolated Signer-Independent Sign Language Recognition

A Skeleton Aware Multi-Modal SLR framework (SAM-SLR) has been proposed by Jiang *et al.* (2021) to leverage additional information beyond a regular camera and thus improve SLR accuracy. The multi-modal architecture encompasses the use of three different data sources, namely pose, RGB and RGB-depth data, to improve SLR performance. Furthermore, Figure 2.1 shows the different architectures combined for each data source to construct the final model. The system uses a multi-stream Sign Language Graph Convolutional Network (SL-GCN), a novel Separable Spatial-Temporal Convolution Network (SSTCN) and a 3D-CNN for SLR on RGB and RGB-depth videos.

[8] A machine learning architecture that combines several other architectures or multiple data sources (RGB, RGB-Depth, and Skeleton) to obtain better results than single-modality algorithms.

[9] A transformer-based model that learns through the outputs of the individual architectures and ground-truth data using all three models.

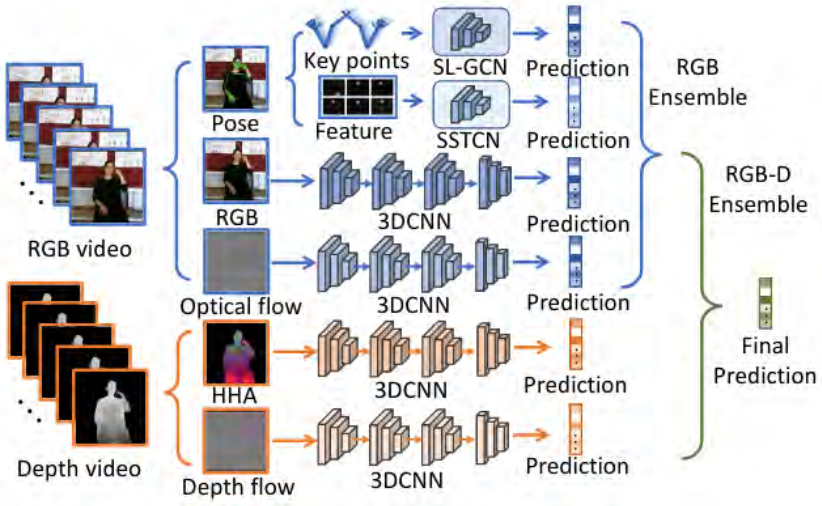


Figure 2.1: Skeleton Aware Multi-modal SLR (SAM-SLR) framework visualising the multi-modal RGB and depth ensemble architecture approach (Jiang *et al.*, 2021).

For the pose stream, two architectures are employed: an SL-GCN and SSTCN. The SL-GCN was combined with an attention mechanism that utilises multi-stream input to extract the motion dynamics of the skeletal graph. Joint coordinates, bone vectors, and joint and bone motion form the multi-stream input. The attention mechanism subsequently extracts motion dynamics from the graph. The network is provided with 133 whole-body landmarks of the signer to construct a spatiotemporal graph. Adjacent spatial dimension points are connected based on natural connections of the human body, and all consecutive landmarks are connected to themselves in the temporal dimension. The node-set is represented by $V = \{v_{i,t} \mid i = 1, \ldots, N, t = 1, \ldots, T\}$ including all 133 landmarks. Subsequently, adjacency matrix $\mathbf{A}$ can be constructed according to Equation 2.2.

$$\mathbf{A}_{i,j} = \begin{cases} 1 & \text{if } d\left(v_i, v_j\right) = 1 \\ 0 & \text{else} \end{cases} \tag{2.2}$$

Where, the minimum distance between skeletal node v_i and v_j is calculated in $d(v_i, v_j)$. Due to a large amount of noise created when utilising 133 nodes, graph reduction was implemented, reducing the skeletal graph to 27 nodes. The 27 nodes consisted of 10 nodes for each hand and seven for the upper body, containing the essential information

of SLR. The second SSTCN pose model further exploits whole-body skeletal features, delving deeper and extracting the nuanced motion patterns inherent in specific signs.

To exploit RGB and RGB-depth data, a 3D-CNN is implemented to compute RGB, optical-flow, HHA and depth flow SLR predictions. The 3DCNN consists of a R(2+1)D-18 convolutional network (Tran *et al.*, 2018), which employs a ResNet backbone with an 18-layer architecture, subsequently splitting the 3D CNN computation into a spatial 2D convolution followed by a temporal 1D convolution, discussed further in Section 2.3.1.3.

A simple ensemble method is applied, combining all the outputs of the pose, RGB and RGB-depth modalities. Each modality is assigned a weight based on the accuracy achieved on the validation set. The modalities are summed up with weights to calculate the final predicted score. The proposed system achieved accuracies of 98.42% and 98.53% on the RGB and RGB-depth AUTSL datasets (Sincan and Keles, 2020), respectively. Outperforming the best baseline CNN[10] model implemented by the original dataset authors, Sincan and Keles (2020), which achieved accuracies of 49.22% and 62.02% for RGB and RGB-depth data, respectively.

De Coster *et al.* (2021) introduce pose flow and self-attention on the AUTSL dataset (Sincan and Keles, 2020) as a technique towards more robust signer-independent SLR framework. A field of 32 frames is accumulated by uniformly selecting 16 pairs of 2 consecutive frames. Non-manual sign-language components are not considered in this study. Hand cropping from the frames is implemented using the OpenPose landmark information. Elbow and wrist landmarks define the region for the square crops, as seen in Figure 2.2.

To account for the loss of movement information due to hand cropping, movement encoding based on optical flow is extracted by utilising the OpenPose landmarks. This is achieved by computing the angle and magnitude for the given landmark vector by the difference in positions of a landmark in two consecutive frames for a selection of K landmarks. The complete feature vector for all landmarks represents the body's movements

[10]The baseline CNN model utilised a hybrid CNN-BiLSTM architecture combined with a feature pooling model after the CNN model.

using fewer dimensions than optical flow.

Figure 2.2: Hand cropping utilising OpenPose wrist and elbow landmarks for more robust cropping of both hands (De Coster *et al.*, 2021).

The following models are analysed in this study for signer-independent SLR:

- VTN, discussed in Section 2.2.1, is applied to the raw RGB inputs.
- VTN with hand cropping as shown in Figure 2.2. This significantly increases the spatial resolution of the hands.
- VTN with hand crops and pose flow. Building on the previous model, pose flow is introduced to the hand crops with 53 body landmarks. A 106-dimensional feature vector is concatenated with the feature vectors extracted by the CNN.

ResNet-34 (He *et al.*, 2016) pretrained on ImageNet (Deng *et al.*, 2009) was chosen as the backbone for the VTN 2D feature extractor. The best-performing model was VTN with hand crops and pose flow, which achieved an accuracy of 92.92% on the AUTSL test set, improving the accuracy of the baseline VTN by 8.10%.

SLR faces a critical hurdle when moving beyond individual signers towards signer-independence. To bridge this gap, researchers have explored innovative approaches, as showcased

in this section. The SAM-SLR framework, proposed by Jiang *et al.*, harnesses multiple data modalities – pose, RGB, and RGB-depth. Consequently capturing the nuances of individual signing styles beyond just hand gestures. Specialized networks like SL-GCN and SSTCN delve deeper, extracting subtle motion patterns and dynamics. This multi-pronged approach results in state-of-the-art accuracies of 98.42% and 98.53% on RGB and RGB-depth AUTSL datasets, significantly surpassing baseline models.

Furthermore, De Coster *et al.* (2021) work with pose flow and self-attention highlights the potential of advanced techniques. By incorporating hand cropping and pose flow encoding, they achieved a 92.92% accuracy on AUTSL, demonstrating the effectiveness of tailored approaches for isolated signer-independent SLR. The quest for robust signer-independent SLR systems holds immense significance. It paves the way for smoother communication between Deaf and Hearing individuals, fostering greater accessibility and inclusivity.

2.2.3 Critical Analysis of the Related Studies

The related studies section covered several studies regarding the different aspects and techniques for addressing isolated SLR and signer-independence. Signer-independent SLR was identified to encounter challenges encompassing signing speed, style variations, as well as disparities arising from left or right-hand dominance. Moreover, distinctions between fluent and natively taught signing styles may also affect meaning through visual-spatial representation variation. Therefore, addressing these issues is crucial to an effective signer-independent isolated SLR system.

SAM-SLR proved successful in Section 2.2.2 on the AUTSL dataset for isolated SLR by implementing a multi-modal framework combining pose, RGB and RGB-depth modalities. The SAM-SLR architecture outperformed the CNN-RNN baseline model proposed by Sincan and Keles (2020) by 49.20% on the RGB AUTSL dataset. The resulting accuracies on the ensemble method of 98.42% and 98.53% on the RGB and RGB-depth datasets, respectively, are impressive. However, it is important to note that only 33 pose landmarks were utilised, which removes significant details such as facial expressions and removal

of subtle visual-spatial representations from signer-independent sign language gestures, which potentially oversimplifies the classification problem.

In Section 2.2.1, Li *et al.* showed as the number of words increases toward real-world SLR vocabulary sizes, the isolated SLR model's accuracy tends to decrease. Accuracy on the smaller WLASL subsets ranged between 70.15% and 89.92% with signer-dependence. In contrast, on the much larger subset consisting of over 2000 words with signer-independence, the best baseline model was only able to achieve an accuracy of 66.31%.

De Coster *et al.* demonstrated that cropping out the hands and introducing pose flow to account for the movement parameter in SLR was successful for signer-independent SLR in Section 2.2.2. The system achieved an accuracy of 92.92% on the AUTSL dataset with a hybrid video transformer combined with a ResNet-34 feature extractor. However, information may be lost as non-manual sign language features are not included in the classification model. De Coster *et al.* also found that the MediaPipe Holistic pose estimation framework outperforms OpenPose and MMPose in performance and computational speed processing by 4.8 FPS. MediaPipe importantly runs on the CPU, which bids well for potential mobile development of SLR systems.

The literature review analysis showed that implementing hybrid RNN models utilising either LSTMs, BiLSTMs, GRU or Transformers combined with feature extraction can be the best approach for optimising accuracy in an SLR system. Furthermore, pose estimation algorithms, CNNs, and vision transformers can be utilised to advance the system. However, pose estimation systems need to incorporate graph convolution networks to better model spatial and temporal information from landmark data. Finally, the pose estimation system should incorporate all relevant landmarks for SLR manual and non-manual features.

2.3 Sign Language Recognition Machine Learning Architectures and Algorithms

Hitherto, it is clear that SLR literature relies extensively on machine learning algorithms, which form part of artificial intelligence. This is important in order for computers to learn and make predictions from sign language data without explicit programming, allowing for the generalisation of unseen data. This section explains the core machine learning algorithms that serve as the foundation for current state-of-the-art SLR methodologies.

Supervised learning is particularly common as the algorithms establish relationships between input features and target labels. Through iterative analysis of large datasets, machines discern patterns, improving performance over time (Cunningham *et al.*, 2008). This iterative learning process significantly enhances the accuracy and efficacy of SLR systems, advancing communication accessibility for the Deaf. SLR typically involves the non-trivial task of interpreting visually moving sign actions for communication – explained in Section 2.1.1. Therefore, it is clear that SLR requires the analyses of both the spatial and temporal dimensions within data using complex deep learning models, as there is a visual aspect combined with a time component for each intricate sign action.

Pose estimation algorithms are appropriate towards action recognition tasks such as SLR by determining spatial positions of key body joints from visual inputs and can be combined with RNNs or GNNs to incorporate the temporal domain – further discussed in Sections 2.3.2 and 2.3.3. Other techniques, such as CNNs and vision transformers, have been proposed for signer-independent SLR to improve the system's robustness and generalisation ability by extraction of relevant features, which are discussed in Section 2.3.1 and 2.3.5. To incorporate the temporal component for SLR, pose estimation algorithms and CNNs are often combined with RNNs – further explained in Section 2.3.4.

2.3.1 Convolutional Neural Networks

CNNs are a specialised class of deep learning models designed primarily for image recognition and processing tasks (Gu *et al.*, 2018). They are inspired by the biological visual cortex's structure and function, automatically allowing them to learn hierarchical patterns and features from raw pixel data. CNNs employ convolutional layers that use small filters to scan input images, extracting local features and creating feature maps. These feature maps are downsampled through pooling layers to reduce spatial dimensions while preserving important information. The extracted features are further fed into fully connected layers for classification or regression. CNNs have revolutionised computer vision by achieving cutting-edge performance in image recognition, object detection, and other visual tasks, making them an integral part of modern artificial intelligence systems (Bhatt *et al.*, 2021).

2.3.1.1 ResNet

He *et al.* (2016) introduced ResNet (Residual Network), a revolutionary architecture that addresses the degradation challenge associated with deep neural networks. The authors employed a deep residual learning framework to mitigate the diminishing performance with increased network depth. This innovative approach incorporates "shortcut connections" or residual links, allowing connections to bypass one or more layers. These shortcut connections are essentially identity mappings, where their outputs are added to the outputs of the stacked layers. This design enables the system to learn residual functions without introducing computational complexity, facilitating easier optimisation and faster computation in deep networks. A visual depiction of the ResNet architecture's residual block utilising a shortcut connection is illustrated in Figure 2.3.

The authors compared the proposed ResNet architecture with well-established deep learning models, including VGG-16, GoogLeNet, and PReLU-net. The evaluation utilised the ImageNet 2012 classification dataset (Russakovsky *et al.*, 2015), consisting of 1000 classes

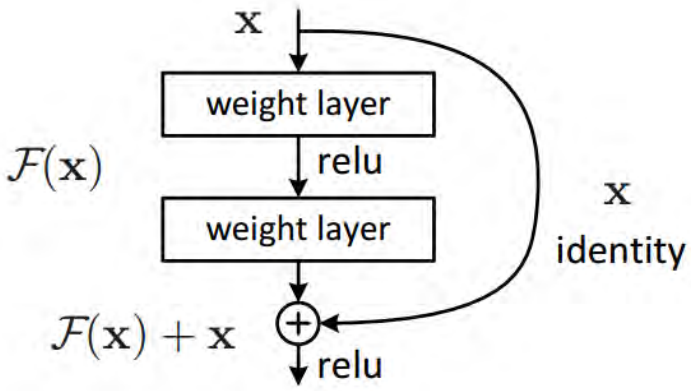


Figure 2.3: The ResNet residual block showing shortcut connection x(He *et al.*, 2016).

and over 1.4 million images. This was split into training with just over 1.2 million images, 50k for validation and 100k for testing.

The comparative analysis revealed that ResNet consistently achieved lower top-5 error rates than the other architectures examined. Even the top-performing existing models exhibited error rates equivalent to the least-performing ResNet model. Notably, ResNet models demonstrated a trend of decreasing error rates with increasing layer depth.

2.3.1.2 Inception

The Inception architecture, also known as GoogLeNet, is a deep learning CNN introduced by researchers at Google in 2014 (Szegedy *et al.*, 2015). It was designed to address some of the limitations of traditional CNNs, such as computational complexity and overfitting, while achieving state-of-the-art performance in image recognition tasks, particularly on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Russakovsky *et al.*, 2015).

The main innovation in the Inception architecture is incorporating Inception modules, which allow the network to efficiently capture multi-scale features and make it deeper without significantly increasing the computational cost. The Inception module utilises multiple parallel convolutional filters of different sizes (e.g., 1×1, 3×3, 5×5) and pooling operations, capturing local and global patterns within the same layer, as depicted in Figure 2.4.

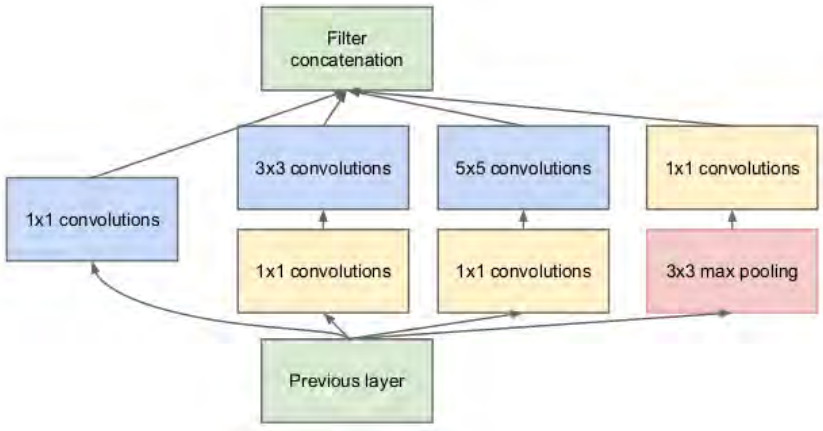


Figure 2.4: Inception module showcasing the multiple parallel convolutional filters (Szegedy *et al.*, 2015).

The Inception module architecture blocks can be explained as follows:

1. 1×1 Convolution: A 1×1 convolutional layer is used to reduce the number of input channels. This helps in reducing computational complexity and allows the network to capture different linear combinations of the input features.

2. 3×3 and 5×5 Convolution: The Inception module also includes parallel 3×3 and 5×5 convolutional layers, capturing different spatial patterns in the input data.

3. MaxPooling: To capture a larger context and reduce spatial dimensions, max-pooling is applied to the input.

4. Concatenation: The outputs of all the different branches (1×1, 3×3, 5×5, and max-pooling) are concatenated depth-wise into a single tensor, preserving their spatial dimensions.

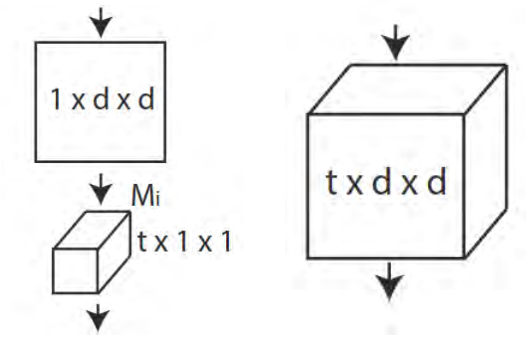
By using these Inception modules in a carefully designed stacked fashion, the Inception architecture can efficiently learn both local and global features robustly, making it effective for various computer vision tasks. The original GoogLeNet model had 22 layers, which was relatively deep at the time, but its efficiency in terms of computational complexity allowed it to be trained on a large scale.

The auxiliary classifiers in GoogLeNet failed to contribute significantly at the end of the training, and the authors argued they function as regularises. Consequently, InceptionV3 integrated batch normalisation in auxiliary classifiers, label smoothing and an RMSProp Optimiser Szegedy *et al.* (2016). One significant improvement was the use of "factorisation" where larger convolutions (e.g., 5×5 and 3×3) were replaced with two consecutive 3×3 convolutions, reducing the number of parameters and enabling more efficient learning. Despite the increased depth, the introduction of factorisation and batch normalisation made InceptionV3 computationally more efficient than its predecessors. These advancements led to significant gains in accuracy and performance, making InceptionV3 one of the prominent choices for various image-related tasks.

2.3.1.3 Spatiotemporal Convolutions

Deep learning with 2D CNNs has driven the field of still-image recognition, particularly the introduction of AlexNet and ResNet (Krizhevsky *et al.*, 2012, He *et al.*, 2016). However, the major drawback with video analysis is that the 2D CNNs cannot model temporal information and motion patterns effectively (Tran *et al.*, 2018). To address the need to model spatial and temporal information in video analysis, 3D CNNs have been widely explored for action recognition tasks, particularly in signer-dependent SLR (Huang *et al.*, 2015). 3D CNNs apply filters (kernels) to 3D input data, sliding the filters along the spatial and temporal dimensions to extract features at each location. This process outputs a set of 3D activation maps that represent the learned features. The activation maps are passed through a series of layers, such as pooling, normalisation, and fully connected layers, to produce the final output.

In comparison, the R(2+1)D spatiotemporal convolutional block has proven successful over 3D ResNets on action recognition tasks (Tran *et al.*, 2018, Hara *et al.*, 2017). An R(2+1)D convolution block applies two separate convolutions to the input data: a 2D spatial convolution and a 1D temporal convolution. The output of the two convolutions is combined to produce the final output. The spatial convolution is applied to the height and width dimensions of the input tensor, while the temporal convolution is applied to the time



(a) (2+1)D Convolution. **(b)** 3D Convolution.

Figure 2.5: Comparison of the (2+1)D convolutional block and 3D convolutional block.

dimension, visualised in Figure 2.5a. This approach allows the R(2+1)D block to capture spatial and temporal features efficiently. In contrast, a 3D convolution applies a single convolution to the input tensor that operates on all three dimensions simultaneously, as seen in Figure 2.5b. The R(2+1)D approach has two advantages: doubling the number of nonlinearities due to the additional ReLU between the 2D and 1D convolutional blocks and making optimisation easier by forcing the 3D convolution into two separate spatial and temporal components.

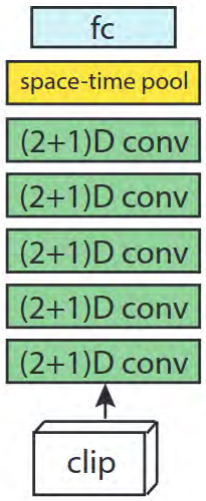


Figure 2.6: R(2+1)D network architecture where the (2+1)D convolutions are ResNets.

The R(2+1)D ResNet architecture consists of a series of ResNet (2+1)D blocks, each containing 2D and 1D convolutional layers and residual connections, as depicted in Figure 2.6. The number of blocks and filters in each block depends on the variant of the

architecture being used.

2.3.2 Pose Estimation

Pose estimation utilises key points on an object or person to infer the pose of the object or person in an image or video (Munea *et al.*, 2020). In people, the key points are generally the major joints in the body. Inferred pose estimations can be performed in either 2D or 3D.

Two popular frameworks for pose estimation are OpenPose and MediaPipe (Cao *et al.*, 2019, Lugaresi *et al.*, 2019).

OpenPose is a state-of-the-art approach to real-time human pose estimation. The OpenPose pipeline takes an RGB image fed into a two-branch multistage CNN. Confidence maps of different body parts are predicted in the first branch. The second branch predicts part affinity fields to encode the orientation and location of limbs. Both branches are processed using bipartite matching to produce 2D key points for pose estimation.

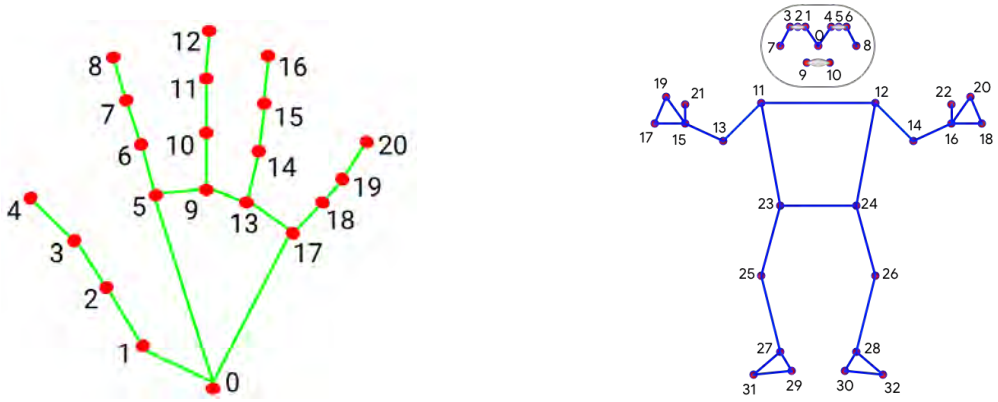
MediaPipe is a framework for building a perception pipeline as a graph of reusable calculators (Lugaresi *et al.*, 2019). Object detection, face landmark, segmentation and hand landmark constraint are a few solutions built using MediaPipe. The MediaPipe Holistic model incorporates human pose, face landmark and hand tracking algorithms. The human pose tracking utilises BlazePose, a lightweight CNN architecture tailored towards real-time human pose estimation (Bazarevsky *et al.*, 2020). BlazePose has two parts towards human pose estimation: an encoder-decoder network architecture to predict heatmaps for all joints, which is used as input for the other part – a regression encoder that produces the coordinates of the joints directly.

2.3.2.1 MediaPipe Hands

Hand pose estimation involves the process of modelling a human hand. Modelling the human hand consists of extracting the parts of the hand, for example, the palm and

fingers, and locating their positions on the hand, also referred to as landmarks (Li *et al.*, 2019).

MediaPipe Hands is a hand tracking solution that predicts hand skeletal data from a single RGB camera (Zhang *et al.*, 2020). The solution consists of two models working together: a palm detector model followed by a hand landmark model. A single-shot detector and oriented hand bounding box detect initial hand locations and crop out the bounding box. The hand landmark model extracts and localises the 21 2.5D landmark coordinates using the cropped hand box. Figure 2.7a visualises these landmarks.



(a) MediaPipe Hands topology. **(b)** BlazePose landmark topology.

Figure 2.7: MediaPipe Pose and Hand topologies (Bazarevsky *et al.*, 2020, Zhang *et al.*, 2020)

Each of the 21 landmarks contains coordinates x, y, and z, where z is the relative depth. The model is robust enough to detect partially visible hands and self-occlusions.

2.3.2.2 MediaPipe Pose

The MediaPipe Pose solution utilises BlazePose, a lightweight CNN architecture for human pose estimation (Bazarevsky *et al.*, 2020). The architecture is implemented in two stages: an initial encoder-decoder framework to predict heatmaps for all joints, which is passed to the second stage encoder that directly regresses the joint coordinates. To overcome the limitations of the Non-Maximum Suppression algorithm, BlazePose assumes that the head of the person is always visible, thus focusing on initially detecting the

bounding box area of the human face. BlazePose outputs 33 landmarks on the human body, each with x, y, and z coordinates, as shown in Figure 2.7b.

2.3.3 Graph Neural Networks

Typically, machine learning tasks are represented in Euclidean space. However, an increasingly new number of applications need to model data from non-Euclidean domains, typically represented through graphs, allowing for the representation of complex relationships and interdependency between objects. Therefore, Graph Neural Networks (GNNs) were introduced, designed to analyse and make predictions on graph-structured data (Wu *et al.*, 2021), leveraging node and edge features and graph topology to learn complex relationships and capture contextual information for improved predictive performance.

One form of a GNN is a Spatiotemporal Graph Convolutional Network (ST-GCN), which takes an undirected graph and computes both spatial and temporal dimensions of the graph. Data in ST-GCNs is represented as a spatiotemporal graph, denoted as $G = (V, E)$, where V represents nodes representing body joints, and E represents edges that link the joints (Feng and Meunier, 2022). The edges are represented as an adjacency matrix A, which denotes whether there is a link between two nodes. The core of ST-GCNs lies in their adaptation of graph convolutional operations to capture spatial and temporal dependencies simultaneously, as visualised in Figure 2.8.

Equation 2.3 defines the adaptive graph convolution operation (Shi *et al.*, 2019).

$$\mathbf{f}_{out} = \sum_{k}^{K_v} \mathbf{W}_k \mathbf{f}_{in} \left(\mathbf{A}_k + \mathbf{B}_k + \mathbf{C}_k \right) \tag{2.3}$$

The adjacency matrix consists of three components: A_k representing the physical structure of the human body and serving as the skeletal graph, B_k being a parameterised adjacency matrix that learns throughout training, allowing the model to focus on recognition tasks and detect significant joints, and C_k forming a distinct graph for each input, determin-

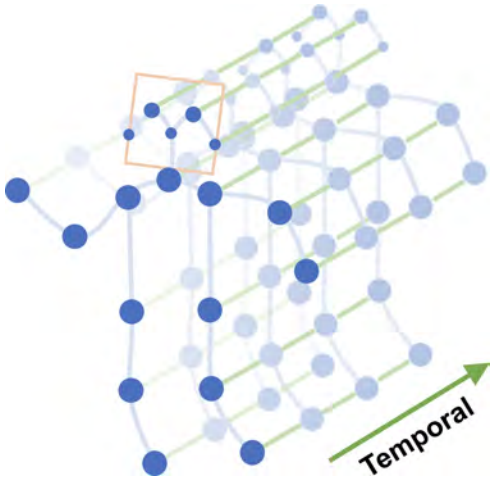


Figure 2.8: The spatiotemporal graph visualising a human skeleton sequence as input to the ST-GCN architecture (Yan *et al.*, 2018). The solid blue dots indicate the body joints with the intra-body edge connection between joints. The temporal inter-frame edges connect the same joints between consecutive frames, represented by the faded blue dots.

ing connections and their strengths through a similarity calculation using a normalised embedded Gaussian function.

ST-GCNs combine spatial and temporal information using graph convolutional operations across both dimensions (Yan *et al.*, 2018). Multiple layers of these operations extract abstract features progressively. Temporal evolution[11] is captured through temporal convolutional layers, typically implemented as 1D convolutions as defined by Equation 2.4.

$$H' = \sigma(Conv1D(H, W_{\text{temporal}})) \tag{2.4}$$

Where H is the node feature matrix and $W_{temporal}$ represents the learnable weights of the temporal convolutional layer.

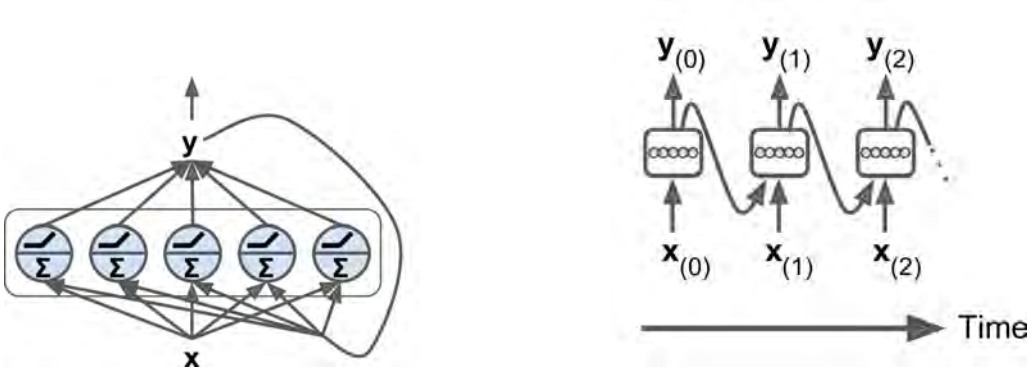
Pooling and aggregation layers are employed for dimensionality reduction and hierarchical feature extraction. ST-GCNs conclude with task-specific, fully connected output layers adapted to the application. ST-GCNs are trained using labelled spatiotemporal

[11]The process of capturing how landmark information propagates and changes over time within the skeletal graph.

data. During training, backpropagation and optimisation algorithms adjust the weights of graph convolutional, temporal, and fully connected layers iteratively. ST-GCNs excel at capturing complex spatiotemporal dependencies in various applications, including action recognition, traffic forecasting, and dynamic social network analysis (Peng *et al.*, 2021).

2.3.4 Recurrent Neural Networks

RNNs are an artificial neural network designed to work with temporal information in data involving sequences. RNNs differ from feed-forward neural networks, containing internal memory and connections pointing backward. A typical RNN consists of layers of recurrent neurons unrolled through time, as depicted by Figure 2.9 (Géron, 2019). The same function is applied to every input of data, while output in the current recurrent neuron layer depends on the previous computation.



(a) Layer of recurrent neurons. **(b)** Recurrent neurons unrolled through time.

Figure 2.9: A layer of recurrent neurons 2.9a unrolled through time 2.9b. Each neuron receives both the input vector $\mathbf{x}_t$ and the output vector from the previous time step $\mathbf{y}_{t-1}$, at each time step (Géron, 2019).

Equation 2.5 illustrates the computation of the output vector for a recurrent neural layer (Géron, 2019).

$$\mathbf{y}_{(t)} = \phi\left(\mathbf{W}_x^\top \mathbf{x}_{(t)} + \mathbf{W}_y^\top \mathbf{y}_{(t-1)} + \mathbf{b}\right) \tag{2.5}$$

Where $\phi(.)$ is the activation function and $\mathbf{b}$ is the bias vector. $\mathbf{W}_x$ and $\mathbf{W}_y$ contain the connection weights of the inputs of the current time step and outputs of the previous

time step, respectively. $\mathbf{x}_{(t)}$ is the input vector, and $\mathbf{y}_{(t-1)}$ is the output vector from the previous step.

As the data goes through many transformations when traversing the RNN, some information tends to be lost at each time step, also known as the vanishing gradient problem (Hochreiter, 1998). To handle this short-term memory problem, various long-term memory cells have been introduced, namely, LSTMs and GRUs (Hochreiter and Schmidhuber, 1997, Cho *et al.*, 2014)

2.3.4.1 Long Short-Term Memory

The LSTM cell architecture consists of several components that interact with each other to store and retrieve information selectively. Figure 2.10 demonstrates the LSTM cell architecture (Géron, 2019).

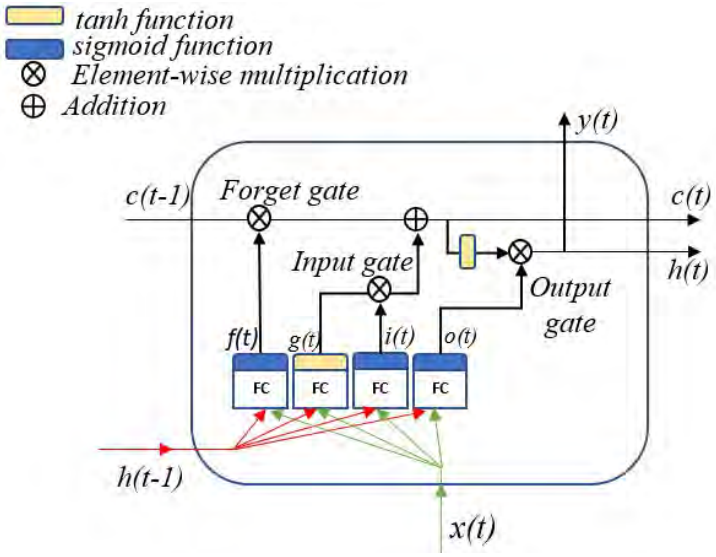


Figure 2.10: Single LSTM memory block depicting the gates and activation functions (Ouassil *et al.*, 2022).

At each time step t, the LSTM cell takes as input the input vector $x(t)$, the previous output vector $h(t-1)$, and the previous cell state vector $c(t-1)$. It computes the output vector $h(t)$ and the cell state vector $c(t)$, which are passed to the next time step (Graves, 2012). The LSTM cell consists of four main components: the input gate, the forget gate, the cell state, and the output gate.

1. Input gate(controlled by $\mathbf{i}_{(t)}$): The input gate decides which information to let into the cell state.

2. Forget gate (controlled by $\mathbf{f}_{(t)}$): The forget gate decides which information to forget from the cell state.

3. Cell state: The cell state represents the internal memory of the LSTM cell. It is updated by combining the information from the input gate and the forget gate as seen in Equation 2.6

$$c(t) = f(t) \odot c(t-1) + i(t) \odot \hat{c}(t) \tag{2.6}$$

   The cell state vector $c(t)$ is passed on to the output gate, selectively determining which part of the cell state to output.

4. Output gate (controlled by $\mathbf{o}_{(t)}$): The output gate decides which information to output from the cell state, both to $\mathbf{h}_{(t)}$ and $\mathbf{y}_{(t)}$. The output gate selectively outputs information from the cell state by element-wise multiplication with the hyperbolic tangent of the cell state, given by Equation 2.7.

$$h(t) = o(t) \odot \tanh(c(t)) \tag{2.7}$$

   Here, tanh is the hyperbolic tangent activation function because it facilitates stable and effective learning of long-term dependencies, and $h(t)$ is the output vector.

Overall, the LSTM cell architecture allows the model to selectively store and retrieve information and learns to recognise important information and store it in the long-term state.

2.3.4.2 Gated Recurrent Unit

The GRU (Cho *et al.*, 2014) is an encoder-decoder framework introduced in RNNs to address the vanishing gradient problem. The GRU consists of two gating units to modulate

the flow of information, an update and a reset gate. The update and reset gates for the j-th element of a vector are calculated according to Equations 2.8 and 2.9, respectively.

$$z_j = \sigma\left(\left[\mathbf{W}_z\mathbf{x}\right]_j + \left[\mathbf{U}_z\mathbf{h}_{\langle t-1\rangle}\right]_j\right) \tag{2.8}$$

$$r_j = \sigma\left(\left[\mathbf{W}_r\mathbf{x}\right]_j + \left[\mathbf{U}_r\mathbf{h}_{\langle t-1\rangle}\right]_j\right) \tag{2.9}$$

Where the logistic sigmoid function is given by σ, $\mathbf{x}$ and $\mathbf{h}_{t-1}$ are the inputs from the previous hidden state. $\mathbf{W}_r$, $\mathbf{W}_z$, $\mathbf{U}_r$ and $\mathbf{U}_z$ are learnt weight matrices.

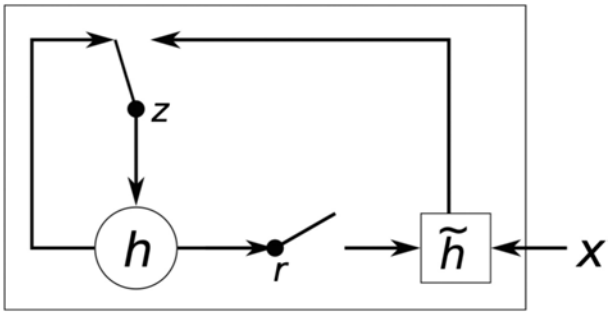


Figure 2.11: Gated Recurrent Unit visualising the update and reset gates in a single unit (Cho *et al.*, 2014).

Figure 2.11 illustrates the GRU update and reset gates where update gate z determines whether the new hidden state $\tilde{h}$ updates the previous hidden state h. The reset gate r decides if the previous hidden state is ignored.

2.3.5 Vision Transformers

Transformers are a type of deep learning model that has gained popularity in natural language processing tasks such as language translation, text generation, and sentiment analysis (Vaswani *et al.*, 2017). The main idea behind transformers is that they use a self-attention mechanism to process input data rather than the traditional RNN or CNN architectures. Self-attention allows the model to focus on different parts of the input sequence and learn which parts are essential for the task.

2.3.5.1 Video Vision Transformer

The Video Vision Transformer (ViViT) is a deep learning model for video recognition that is based on the Vision Transformer (ViT) architecture (Dosovitskiy *et al.*, 2020). ViViT consists of three main components: a spatial encoder, a temporal encoder, and a classification head. The spatial encoder processes each input video frame independently and encodes it into a fixed-length feature vector using a ViT-based transformer network. This is done using a set of 2D position embeddings and a multi-head self-attention mechanism to capture spatial relationships between different regions in the frame.

The output of the spatial encoder is a tensor of size $[T, H, W, C]$, where T is the number of frames, H and W are the height and width of the frame, and C is the number of channels in the feature maps. The temporal encoder takes the feature vectors from the spatial encoder and models their temporal relationships using a 1D temporal convolutional network.

Tubelet embedding efficiently encodes spatiotemporal information of video segments called tubelets. Tokens are extracted from non-overlapping tubelets consisting of dimensions $t \times h \times w, n_t = \left\lfloor \frac{T}{t} \right\rfloor, n_h = \left\lfloor \frac{H}{h} \right\rfloor$ and $n_w = \left\lfloor \frac{W}{w} \right\rfloor$ from the temporal, height and width dimension respectively, as visualised in Figure 2.12. The tubelet embedding module consists of a spatial encoder and a temporal encoder. The spatial encoder uses a modified version of the ViT-based transformer network to encode the appearance features of each frame in the tubelet (Dosovitskiy *et al.*, 2020). The temporal encoder aggregates the appearance features of the frames in the tubelet over time and captures the motion information using a 1D dilated CNN. The output of the tubelet embedding module is fed into the temporal encoder of the ViViT model, which further aggregates the information across multiple tubelets to produce a final prediction.

ViViT achieves state-of-the-art performance on several benchmark video recognition datasets while being significantly more computationally efficient than existing approaches (Khan *et al.*, 2022). This is achieved through architectural optimisations, such as depth-wise convolutions and attention masking, and training strategies, such as distillation and knowledge distillation.

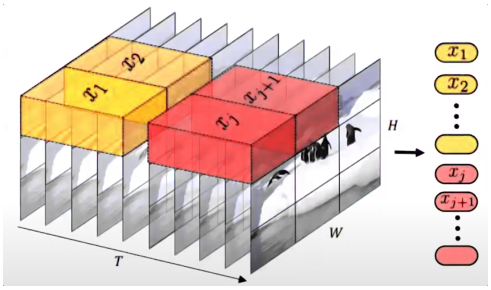


Figure 2.12: An overview of the Tubelet Embedding module with dimensions $t \times h \times w$.

2.3.5.2 Video Swin Transformer

The Video Swin Transformer introduces an inductive locality bias that emphasizes local information processing, optimising its ability to analyze and understand nearby data points within the video sequence, thereby achieving an improved balance between speed and accuracy (Liu *et al.*, 2022). The approach contrasts with earlier methods that globally compute self-attention, even when employing spatial-temporal factorisation, such as in the ViViT model. Video Swin Transformers enhance the performance of ViViT by exploiting the built-in spatiotemporal locality of videos. In this context, pixels that share a closer spatiotemporal distance are more likely to demonstrate correlation. The proposed video architecture introduces locality by adapting the Swin Transformer (Liu *et al.*, 2021) originally designed for the image domain while concurrently harnessing the capabilities of pretrained image models.

The computation of local attention within non-overlapping windows entails reformulating the original Swin Transformer's shifted window mechanism to accommodate spatiotemporal input. This is achieved by replacing multi-head self-attention modules from a standard transformer model with the 3D shifted window-based multi-head self-attention module, which is visualised in Figure 2.13.

In the Video Swin Transformer 3D shifted window architecture, the input size is $8 \times 8 \times 8$, and the 3D window size is $4 \times 4 \times 4$. Each layer of the network uses regular window partitioning, which means that each layer divides the input into a set of non-overlapping windows. In layer l, the number of windows is equal to $2 \times 2 \times 2 = 8$. The input volume is divided into eight windows, each with a size of $4 \times 4 \times 4$ tokens. The windows in layer

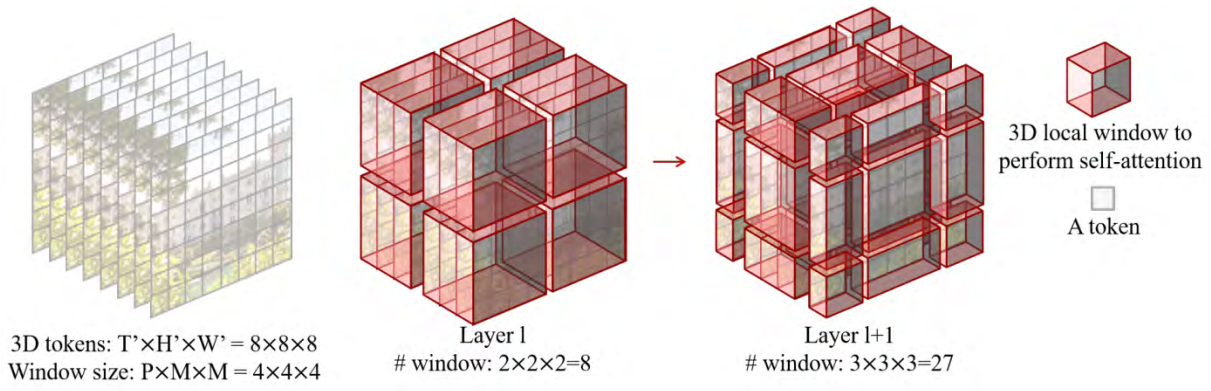


Figure 2.13: 3D shifted windows within the Video Swin Transformer.

$l+1$ are shifted by $(\frac{P}{2}, \frac{M}{2}, \frac{M}{2}) = (2, 2, 2)$ tokens. Subsequently, each window in layer l is shifted by two tokens in each dimension. As a result, the number of windows in layer $l+1$ increases to $3 \times 3 \times 3 = 27$.

Although the number of windows has increased, the efficient batch computation technique described in Liu *et al.* (2021) can still be used, such that the final number of windows for computation is still eight.

2.3.6 Model Regularisation

Model regularisation is a technique in machine learning that aims to prevent overfitting and improve the generalisation of a model to unseen data. Overfitting occurs when a model learns the training data too well, including its noise and fluctuations, to the extent that it performs poorly on new, unseen data (Ying, 2019). Instead of penalising the loss function, regularisation techniques shape the model's training experience, encouraging robust representations and better performance on unseen data.

2.3.6.1 Data Augmentation

CNNs have demonstrated exceptional performance with computer vision tasks. Nevertheless, their effectiveness is closely tied to the availability of extensive datasets to mitigate overfitting. Data augmentation comprises a range of techniques to augment the size and

quality of training datasets (Shorten and Khoshgoftaar, 2019). This augmentation process ultimately contributes to more robust models with better generalisation capabilities. Augmentation of video frames is applied in the same manner as image augmentation, where the same transformation is applied to each frame within the video clip. The augmentations include geometric transformations and colour transformations.

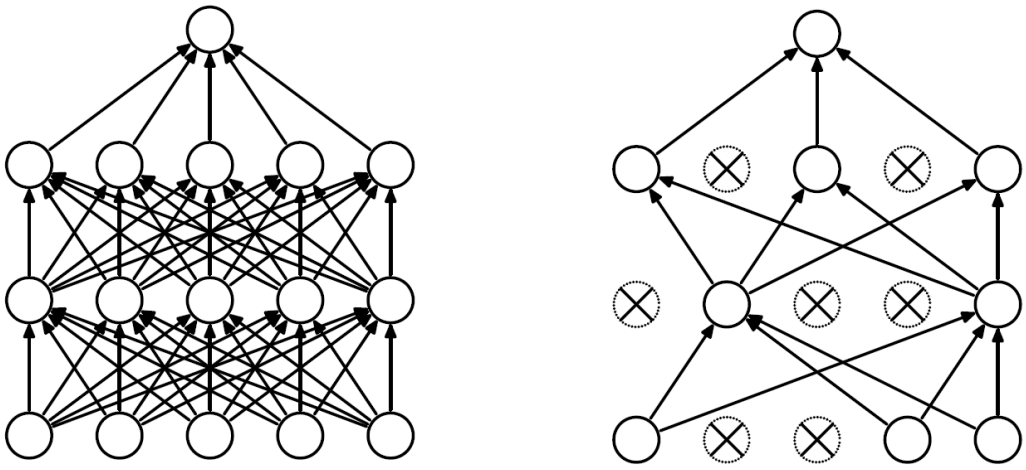
Geometric transformations consist of operations that alter the video frames based on a coordinate system. These operations include random rotations, flips, translations, sheers and zoom within a specified range and given a certain probability. Colour augmentations involve manipulating the colour characteristics of images, encompassing adjustments such as altering brightness, contrast, saturation, or introducing colour-based variations to enhance the diversity of the training dataset.

Carefully choosing appropriate augmentation techniques for a dataset proves advantageous in mitigating potential biases within the data. Yet, an excessively enthusiastic application of augmentation parameters may lead to excessive noise, ultimately producing suboptimal classification results.

2.3.6.2 Dropout

Dropout is a regularisation technique employed in machine learning models, notably within neural networks, to mitigate the risk of overfitting and enhance the generalisation capabilities of the model (Hinton *et al.*, 2012). This technique entails the stochastic deactivation of a random subset of neurons during the training process, effectively excluding their outputs for a given iteration.

Dropping a unit from the network involves temporarily excluding it and all associated incoming and outgoing connections, as illustrated in Figure 2.14. The selection of units to drop is conducted randomly. In the most straightforward scenario, each unit is maintained with a predetermined fixed probability p independent of other units. This probability can be determined through a validation set or set at a fixed value such as 0.5, which has demonstrated efficacy across a range of networks (Srivastava *et al.*, 2014). Figure 2.14b



(a) Standard Neural Network with two hidden layers.

(b) Neural Network after applying dropout.

Figure 2.14: Visualisation of Neural Network before and after the application of dropout regularisation (Srivastava *et al.*, 2014). Dropped neurons are represented with crosses.

is the "thinned" network as a result of sampling the original network made up of the surviving neurons after dropout.

2.3.6.3 Batch Normalisation

Ioffe and Szegedy (2015) developed Batch Normalisation, enabling models to utilise higher learning rates, increase convergence speed, avoid overfitting and reduce the need for dropout layers. Batch Normalisation aims to alleviate the issue of *internal covariate shift*, which refers to the fluctuation in the distribution of layer inputs during training caused by updates to preceding layers (Shimodaira, 2000). The continual variation in input distributions is generally detrimental to model training.

To address this, Ioffe and Szegedy (2015) proposed minimising internal covariate shift by introducing additional network layers specifically designed to normalise the inputs of intermediate layers. Normalising the input data to have a zero mean and a constant standard deviation is a widely accepted, significantly improving model convergence speed (LeCun *et al.*, 1998, Wiesler and Ney, 2011). This notion extends to the intermediate layers in a neural network. The process of linearly transforming data to achieve zero means and a constant standard deviation is referred to as whitening. Consequently, Batch

Normalisation attempts to whiten the activations at specific intervals within a network, mitigating the effects of internal covariate shift and promoting more stable and efficient training.

2.3.6.4 Weight Decay

Weight decay, commonly incorrectly referred to as L2 regularisation, assumes a pivotal role in model regularisation and is proficient at mitigating overfitting. While L2 regularisation and weight decay share a common objective, namely the penalisation of substantial weights to prevent overfitting, they differ conceptually. L2 regularisation entails adding a penalty term to the loss function proportional to the sum of the squared magnitudes of model parameters. Conversely, weight decay achieves a similar outcome by directly modulating the weight updates during optimisation, typically involving the subtraction of a fraction of the weights.

In standard stochastic gradient descent, L2 regularisation and weight decay are frequently treated interchangeably, as both introduce a regularisation term to the loss function. However, in the context of adaptive gradient algorithms such as Adam and AdamW, distinctions emerge (Kingma and Ba, 2014, Loshchilov and Hutter, 2019). AdamW explicitly treats weight decay as a distinct component influencing the optimisation process, particularly by applying weight decay directly to the learning rate. This distinction assumes significance, as empirical studies suggest that AdamW, with its explicit treatment of weight decay, may yield improved stability and generalisation (Loshchilov and Hutter, 2019). Consequently, thoughtful consideration of the interplay between L2 regularisation, weight decay, and the selection of optimisation algorithms is imperative for attaining optimal model performance and generalisation across diverse applications.

2.3.6.5 Early Stopping

Early stopping serves as a strategy to mitigate overfitting by discontinuing model training once specific conditions are satisfied (Prechelt, 1998). The common criterion applied is

the absence of improvement in validation results during training. If the validation results fail to show enhancement within a designated number of training iterations, the training process is prematurely halted. Notably, leveraging early stopping for model regularisation brings the additional benefit of reducing computational complexity (Prechelt, 2012).

2.4 Summary

This chapter analysed related studies and expanded on machine learning concepts and algorithms. The fundamental understanding of SLR syntax and the concept of signer-independence were explained, necessitating the need for signer-independent SLR systems.

Different approaches towards isolated SLR were investigated for comparison purposes. Based on the literature, two successful approaches toward achieving isolated signer-independent SLR exist, namely pose-based and vision-based. Pose-based approaches utilise pose estimation algorithms such as MediaPipe or OpenPose to extract landmarks and learn the feature patterns from the landmarks spatiotemporal GCNs have been proposed.

With regard to the vision-based approaches, traditional models consist of hybrid CNN-RNN models to work with spatial and temporal data. However, the R(2+1)D spatiotemporal convolutional architecture tends to outperform the hybrid approach with action recognition tasks, which is ideal for SLR. The latest vision-based approach is video vision transformers, which are transforming the field of computer vision and have the ability to deal with longer-term dependencies in temporal data.

Subsequently, these potential pose-based and vision-based approaches are explored further as unimodal architectures towards isolated signer-independent SLR in Chapter 3.

3

Methodology and Implementation

This chapter explains the decisions towards implementing the various architectures that address isolated signer-independent SLR effectively. First, an overview of the proposed approach and a breakdown of the architecture selection are provided in Section 3.1. Implementation details are subsequently discussed and the signer-independent SLR system application is presented in Section 3.2.

3.1 Methodology

This section provides a high-level overview of the proposed system for signer-independent SLR. The architecture selection of pose-based and vision-based methodologies is explained. Datasets that appropriately capture the signer-independent SLR are also illustrated.

3.1.1 Methodology Overview

Constructing a system for isolated signer-independent SLR requires selecting and analysing the most effective and efficient architecture that can generalise well to different signers as well as signing conditions. Exploring pose estimation with GCNs, hybrid CNN-RNNs, and vision transformers aims to develop a robust system for effective signer-independent SLR, aligning with the objectives set out in Section 1.5.

Two separate pose-based and vision-based approaches are considered for signer-independent SLR. The pose-based approach utilises pose estimation landmarks combined with a GCN,

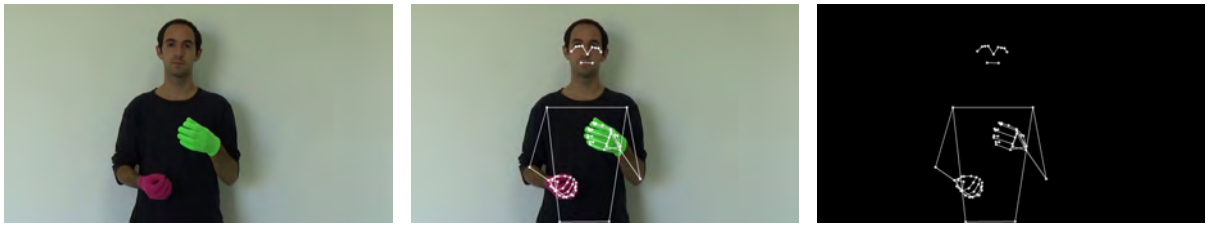
enhancing spatial and temporal between landmarks. In contrast, the vision-based framework compares different CNN and vision transformer approaches for SLR. The first baseline CNN model creates a hybrid CNN-RNN model by combining a CNN model with an RNN model for spatial and temporal data processing, explained in Section 3.1.2.3. Moving beyond the sequential processing limitations of CNN-RNNs, the recently established R(2+1)D-18 and Swin3D-T architectures, which leverage dedicated built-in layers to achieve the concurrent extraction of spatial and temporal features, are implemented as part of the vision-based approaches. The various models are trained to evaluate and compare their performance for signer-independent SLR on the Argentinian LSA64 and Turkish AUTSL datasets.

3.1.2 Architecture Selection

Optimal results in model training necessitate carefully choosing an architecture that can accurately extract relevant features from the dataset. This is vital for successfully classifying distinct signs that remain consistent across different signers. The architectural decisions of the pose-based and vision-based approaches are explained in the following subsections.

3.1.2.1 Pose Estimation

The skeletal-based model utilises the MediaPipe Holistic framework, as discussed in Section 2.3.2 to extract body landmarks. MediaPipe was chosen over other pose estimator frameworks such as OpenPose and MMPose due to its dedicated hand pose estimation model and processing of 4.8 FPS compared to OpenPose 1.1 FPS and MMPose 1.3 FPS (De Coster *et al.*, 2023). The lower extremities are excluded because SLR only requires pose points from the torso and upper extremities. Therefore, 21 landmarks from each hand were combined with 17 pose landmarks from the body to create a total of 59 landmarks, as visualised in Figure 3.1.



(a) Original Image from the LSA64 dataset.

(b) Annotated with MediaPipe Holistic landmarks.

(c) MediaPipe Holistic landmarks only.

Figure 3.1: MediaPipe Holistic visualisation of landmarks.

Based on the study by De Coster *et al.* (2023), the frame landmarks were normalised and missing ones were imputed in the post-processing stage before training.

3.1.2.2 Graph Convolutional Networks

The chosen architecture for analysing the effectiveness of GCNs for signer-independent SLR is the ST-GCN architecture, explained in Section 2.3.3. The ST-GCN architecture is based on Yan *et al.* (2018)'s work, which utilised nine layers of spatial-temporal convolution blocks employing the ResNet mechanism. LSTMs and 1D CNNs fail to model the spatial and temporal domains. In contrast, ST-GCNs undirected graph networks use pose estimation to model body-joint data to represent the inter-body connections between the human joints successfully (Ahmad *et al.*, 2021). Consequently, this enables leveraging more meaningful and useful information for pose-based video classification tasks.

3.1.2.3 CNN Architectures

Two approaches to CNN architectures were considered: hybrid CNN-RNN and spatiotemporal convolutional network. Two popular CNN architectures were analysed for the hybrid model, InceptionV3 and ResNet. They were thus combined with an LSTM RNN architecture to create hybrid CNN-RNN models. The spatiotemporal convolution utilised an R(2+1)D-18 model with a ResNet18 backbone, explained in Section 2.3.1.3. CNN architectures have proven to be highly effective in image and video analysis tasks due to their

ability to capture spatial hierarchies, patterns within the data, translation invariance, and temporal information, as explained in Section 2.3.1.

3.1.2.4 Vision Transformers

The transition from CNNs to ViTs in video classification is motivated by ViTs' capacity to capture global context and relationships between image patches through self-attention mechanisms, allowing them to process spatial and temporal information in videos efficiently (Selva *et al.*, 2023). ViTs offer adaptability across tasks, as they can be pretrained on large image datasets and apply transfer learning for specific video classification tasks. The efficacy of convolution-based methods is constrained by the relatively limited receptive field of the convolution operator. The self-attention mechanism enables the expansion of this receptive field without significantly increasing the number of parameters, in contrast to traditional convolutional methods, which require larger kernels or deeper architectures to achieve similar receptive fields (Liu *et al.*, 2022). The introduction of this mechanism results in improved performance of ViTs for video recognition tasks.

Subsequently, the Video Swin Transformer is proposed for signer-independent SLR over the ViViT vision transformer due to the introduction of an inductive bias of locality and availability of pretrained weights on the Kinetics-400 dataset. The tiny version, Swin3D-T, of the Video Swin Transformer, is applied for signer-independent SLR due to the size of the datasets to avoid overfitting through a smaller model.

3.1.3 Dataset Selection

Two datasets were considered to analyse signer-independent SLR: the Argentinian LSA64 (Ronchetti *et al.*, 2016) and the Turkish AUTSL dataset (Sincan and Keles, 2020). The LSA64 dataset was chosen for its size as a smaller SLR dataset for preliminary experimentation on signer-independence. The fundamental basis behind the LSA64 dataset was the number of signers, which was ten, and the even balance between the number of samples performed by each signer.

The *ChaLearn 2021 CVPR Challenge* was a computer vision challenge focused on isolated signer-independent SLR (Sincan *et al.*, 2021). The goal of the challenge was to encourage the development of new algorithms and techniques for isolated SLR that can perform well on large-scale datasets with signer-independent, real-world variations. The AUTSL comprises 226 signs performed by 43 different signers, with 38,336 isolated sign video samples (Sincan and Keles, 2020). The videos were captured using multiple modalities, including RGB and depth cameras, and annotations were provided for the signing gestures and their corresponding glosses. The dataset includes challenging aspects such as variations in signing style, different lighting conditions, and occlusion. Subsequently, the AUTSL dataset was chosen alongside the LSA64 dataset to gauge the final model performance and compare it to existing studies. The SASL was not considered due to a lack of data, signer-independence and benchmark studies to compare results.

3.2 Implementation

The Implementation section details the application of the specific architectures and data-preprocessing techniques. Furthermore, video dataset sampling, data augmentation, and early stopping are explained in detail.

The high-level overview of the system is shown in Figure 3.2, detailing the systematic SLR process. The model builder phase instantiates the selected SLR architecture. If the pose-based approach is utilised, landmarks are generated and post-processed. The alternative vision-based approach utilises the RGB video input data for training. The models are evaluated on both datasets using isolated train, validation, and test splits. Video dataset sampling involves augmenting the training data and feeding all dataset splits into a video dataset sampling class that uniformly samples the frames from each video. Finally, the model is trained and evaluated to produce results that are analysed and compared to the other architectures implemented in this study and existing SLR architectures.

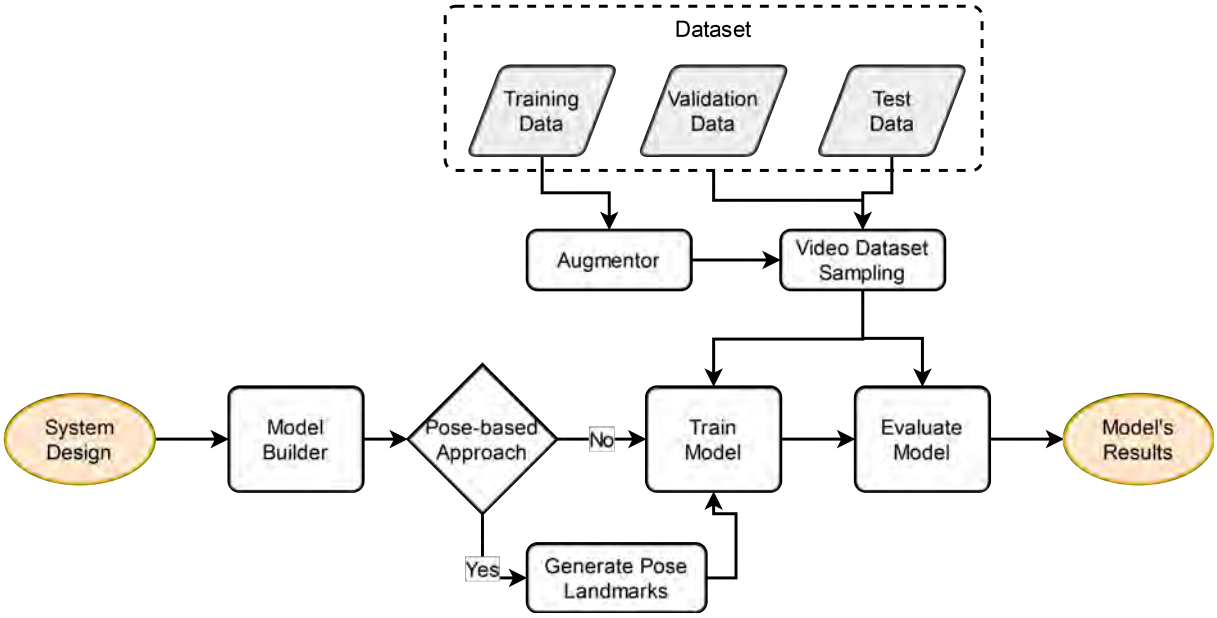


Figure 3.2: High-level systematic overview of the SLR system.

3.2.1 Architectures

The two approaches toward signer-independent SLR are divided into the pose-based and the vision-based approaches, whereby the first approach utilises pose estimation landmarks, and the second utilises RGB frames for signer-independent SLR. The selected architectures, as discussed in Section 3.1.2, are subsequently implemented as follows:

3.2.2 Pose-based Approach

The ST-CGN model combines the MediaPipe Holistic pose estimation framework with the GCN architecture discussed in Sections 3.1.2.1 and 3.1.2.2. To construct this model, the pose estimation landmarks must first be generated and then post-processed to deal with video frames with missing landmarks and subsequently normalised, as depicted in Figure 3.3. An adjacency matrix is generated to show graph connections between the human body's landmarks.

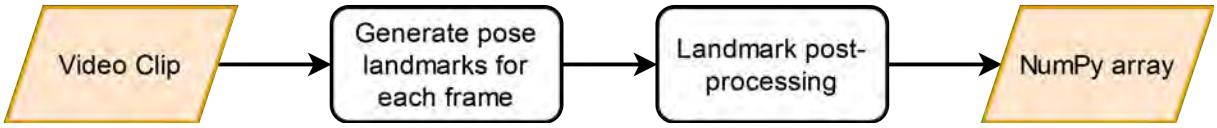


Figure 3.3: Video clip pose landmark generation with subsequent imputation for each landmark subset. The final output is a NumPy array for each video frame.

3.2.2.1 Generate Pose Landmarks

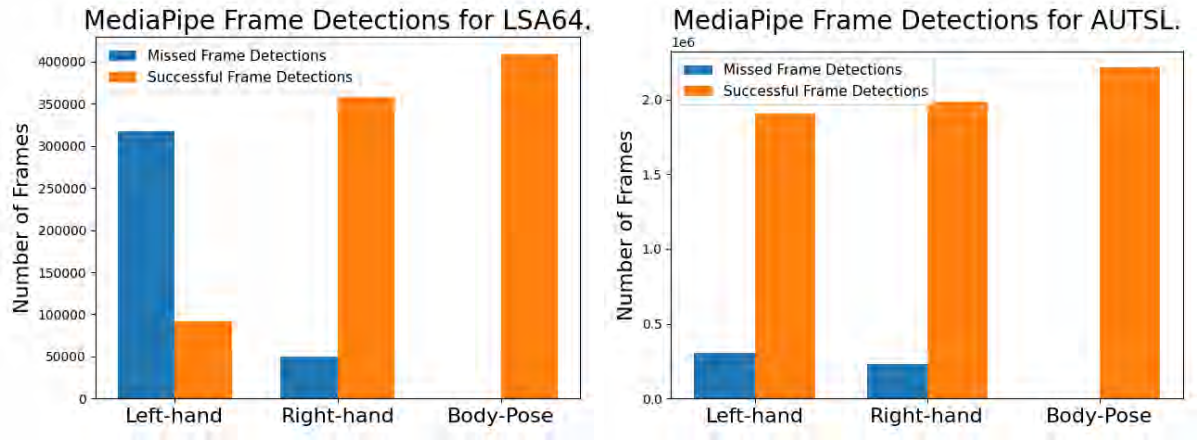
The skeletal-based model utilises the MediaPipe Holistic framework, as discussed in Section 3.1.2.1 to extract body landmarks. The lower extremities are excluded because SLR only requires pose points from the torso and upper extremities. Therefore, 21 landmarks from each hand are combined with 17 pose landmarks from the body to create a total of 59 landmarks. MediaPipe returns each subset of landmarks for the left-hand, right-hand and whole-body pose as a dictionary type. The coordinates returned by MediaPipe are already normalised to $[0.0, 1.0]$ by the image height and width, respectively.

Subsequently, the dictionary type is converted to a NumPy array for each subset containing 21 landmarks for each hand and 17 whole-body landmarks. Each landmark contains three coordinates x, y, z, where z is relative depth. Accordingly, the three subsets are concatenated into a single NumPy array of shape $[59, 3]$ for each frame. For an entire video, the shape is of size $[T, 59, 3]$, where T is the number of frames in the video. The entire video sequence of pose landmarks passed to the post-processing stage to deal with frames with missing landmarks, which is explained in Section 2.3.2.

3.2.2.2 Landmark Post-processing

As elaborated in Section 2.3.2.2, the MediaPipe Holistic detection model either returns all landmark points for each subset of the right-hand, left-hand or whole-body pose or none at all. Figure 3.4 compares the failed and successful detections for the three body subsets for each frame across both datasets.

Instead of dropping the rows of missing frames, an imputation-based approach is adopted to fill in the missing landmarks. The imputation consists of linear interpolation, zero-



(a) LSA64 failed frame detections. **(b)** AUTSL failed frame detections.

Figure 3.4: Comparison of failed frame detections for the LSA64 and AUTSL datasets with MediaPipe Holistic.

based imputation, and backwards fill imputation. Imputation for MediaPipe Holistic detections is performed separately for each subset of left-hand, right-hand and pose landmarks. Linear interpolation is utilised to impute missing frame landmarks using the nearest non-missing previous and subsequent frame landmarks. If no previous frame data exists, backwards filling imputation is exercised, which copies over missing landmarks from subsequent frames. If landmark detection for the entire video sequence of the subset of landmarks fails, zero-based imputation is carried out. This could be scenarios where the left or right hand is not present in the video, a common occurrence within the LSA64 dataset for the left hand, as seen in Figure 3.4a. More detail on the dominant hand and sign breakdown is provided in Section 4.2.

3.2.2.3 Adjacency Matrix

Graphs provide a powerful way to model relationships between entities, and one common representation is through adjacency matrices (Wu *et al.*, 2019). This method particularly applies to human pose graphs, where the connections between body joints or landmarks play a crucial role. An adjacency matrix is built to represent the graph connections from the MediaPipe Holistic landmarks. As the connections between body joints never change, a spatial graph from the landmark can be created. This uses the landmarks as nodes

while defining the set of edges to connect the nodes. As all the landmarks are stored in a single NumPy array, indexes refer to the different subsets where indexes 0-20 belong to the left hand, indexes 21-41 represent the right hand, and indexes 42-58 belong to the pose landmarks. Figure 3.5 visualises the landmarks in the first row and subsequently adds the connections based on the adjacency matrix of edge connections.

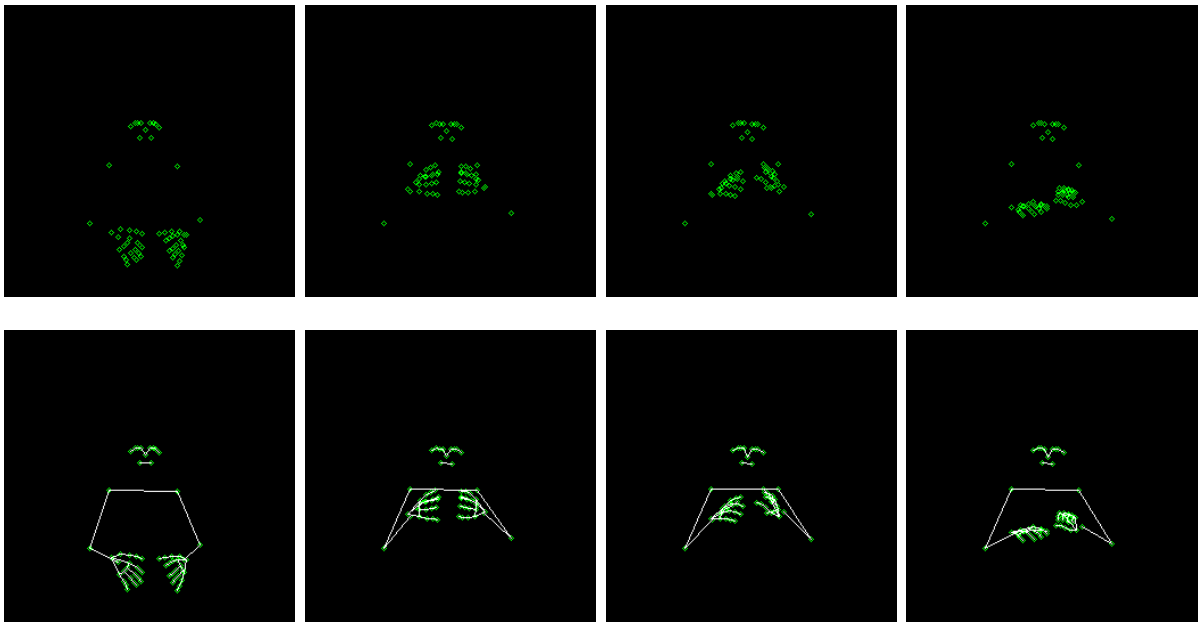


Figure 3.5: Comparison of sample video frames with and without graph connections. The first row depicts the landmarks without any connections, whereas the second row adds the connections from the adjacency matrix.

3.2.2.4 ST-GCN

The ST-GCN model is designed to handle graph-structured data across both spatial and temporal dimensions. The model consists of ten layers or ST-GCN blocks. The first four layers output 64 channels, the second three layers output 128 channels and the final three output 256 channels, as seen in Figure 3.6.

`ST_GCN_block` class in Listing A.1 combines the graph convolution and temporal convolution operations. The adaptive graph convolution layer, `GraphConvolution`, takes in a tensor of shape $C \times T \times N$, where C defines the number of channels, T denotes the number of frames, and N is the number of vertices. The spatial features are extracted using

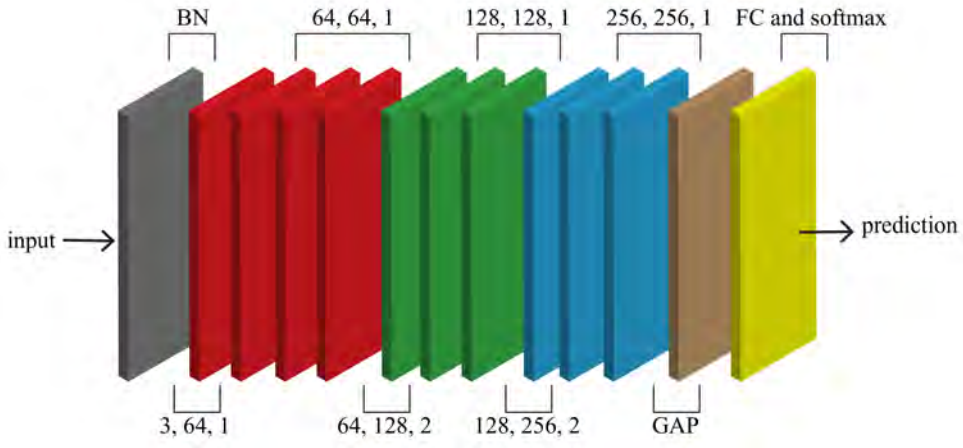


Figure 3.6: ST-GCN architecture (Ghosh *et al.*, 2022).

the adaptive graph convolution layer explained in Section 2.3.3 with a kernel size of three. A 2D convolution layer with $k_t \times 1$ kernel size is applied for the `TemporalConvolution` layer.

The kernel utilises a corresponding stride of 1, meaning the kernel is shifted by one video frame after each iteration. Both the graph and temporal layers undergo batch normalisation and a `ReLU` activation function before finally being concatenated. If residual connections are enabled, a residual block is added to match the dimensions using a 1×1 temporal convolution.

3.2.3 Vision-based Approaches

The vision-based approaches include hybrid CNN-RNN models, spatiotemporal convolutional networks and a Video Swin Transformer as detailed in Section 3.1.2. The implementation of the selected vision-based architectures is specified in the following sub-sections.

3.2.3.1 ResNet50-LSTM

The ResNet50-LSTM model is the first baseline hybrid CNN-RNN model that utilises the ResNet50 backbone to extract spatial features from a video clip and the LSTM architecture to apply temporal processing to the extracted features. The ResNet50 backbone

consists of 50 layers, utilising 3×3 convolutional filters, and is characterised by residual blocks, explained in Section 2.3.1.1. These blocks contain shortcut connections, enabling the direct flow of information across layers and addressing the vanishing gradient issue. Batch normalisation and ReLU activation functions are applied after each convolutional layer, and the network ends with global average pooling and a fully connected layer for classification. The ResNet50 backbone utilises transfer learning by loading the weights of the model trained on the ImageNet-1K[1] dataset.

The LSTM architecture is defined with an input size variable `num_features` from the ResNet50 backbone, a hidden layer size of 128, and two stacked LSTM layers, as seen in Listing A.2. The ResNet50 backbone is combined with the LSTM model to create the hybrid model as seen in Listing A.2, where the extracted features from the ResNet50 backbone are fed into the LSTM and subsequently classified using a fully connected layer.

3.2.3.2 InceptionV3-LSTM

Expanding from Section 3.2.3.1's ResNet50-LSTM, a hybrid InceptionV3-LSTM model is introduced for further analysis of feature extraction impact. InceptionV3 comprises an intricate architecture featuring Inception modules with parallel branches of 1×1, 3×3, and 5×5 convolutions, along with pooling, allowing it to capture features at multiple scales as explained in Section 2.3.1.2. The network incorporates reduction blocks for spatial dimension reduction and auxiliary classifiers for gradient flow, culminating in global average pooling and fully connected layers for image classification. The transfer learning is also applied to the InceptionV3 backbone using the ImageNet-1K dataset.

The LSTM architecture is defined with an input size `num_features` from the final layer output features of the InceptionV3 architecture, a hidden layer size of 128, and two stacked LSTM layers. The InceptionV3 backbone is combined with the LSTM model to create the hybrid model as seen in Listing A.3, where the extracted features are fed into the LSTM and subsequently classified using a fully connected layer.

[1]ImageNet dataset with a 1000 classes (Russakovsky *et al.*, 2015)

3.2.3.3 R(2+1)D-18

The R(2+1)D convolution block applies two separate convolutions to the input data: a 2D spatial convolution and a 1D temporal convolution. The output of these two convolutions is combined to produce the final output. The spatial convolution is applied to the height and width dimensions of the input tensor, while the temporal convolution is applied to the time dimension, as explained in Section 2.3.1.3. This approach allows the R(2+1)D block to capture spatial and temporal features efficiently.

The R(2+1)D-18 model architecture consists of 18 layers based on a ResNet-18 backbone, including 17 convolutional layers and one fully connected layer. The model takes in an input tensor of size (T, C, H, W), where T is the number of frames in the video, C is the number of channels (usually 3 for RGB videos), H is the height of each frame, and W is the width of each frame. The `BasicBlock` for each R(2+1)D layer is detailed in Listing A.4 consisting of two 3D convolutional layers with batch normalisation and ReLU activation in between.

The `Conv1` has a kernel size of (1, 3, 3), implying a size of one along the temporal dimension and 3×3 along the spatial dimension. Therefore, `Conv1` has a kernel size of (3, 1, 1) focusing on the temporal dimension with a size of three and 1×1 along the spatial dimensions. The block also includes a shortcut connection applied if the input size or stride does not equal the output size or stride, involving a 3D convolutional layer and batch normalisation. When implementing the model, transfer learning is applied by loading the weights of the Kinetics-400 dataset (Kay *et al.*, 2017). Hence, the final linear layer output feature is edited to represent the number of classes in the utilised dataset.

3.2.3.4 Swin3D-T

The Swin3D-T is the tiny version of the Video Swin Transformer variants consisting of four stages, depicted in Figure 3.7. The initial input is a video tensor defined by size $T \times H \times W \times 3$ containing T frames. Each 3D path of size $2 \times 4 \times 4 \times 3$ is treated as a token; subsequently, the 3D patch partitioning layer obtains $\frac{T}{2}\frac{H}{4}\frac{W}{4}$ 3D tokens creating a

96-dimensional feature for each patch/token. The features are projected into an arbitrary dimension denoted by C using a linear embedding layer. Within the patch merging layers, $2\times$ spatial downsampling is performed, concatenating the features of each 2×2 group of spatially neighbouring patches.

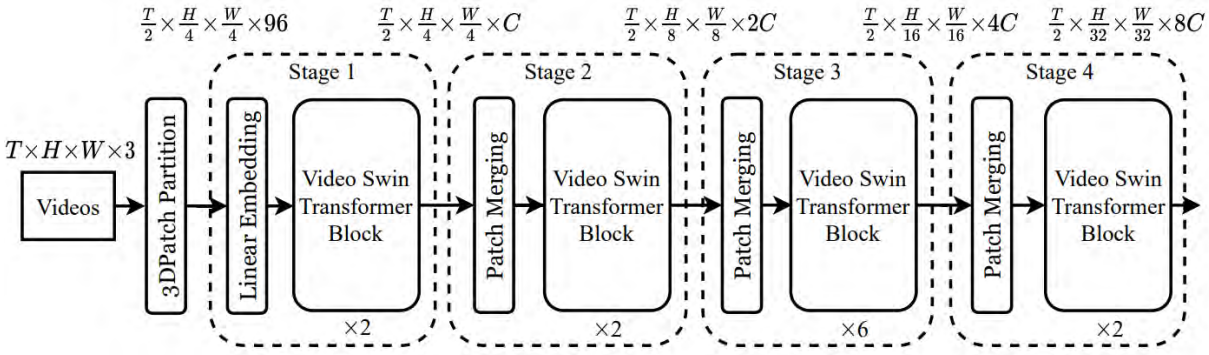


Figure 3.7: Architecture of the tiny version (Swin3D-T) of Video Swin Transformer.

The Video Swin Transformer block applies the 3D shifted window-based multi-head self-attention model, as explained in Section 2.3.5.2 while keeping the other components of the original transformer layer unchanged. Following each Video Swin Transformer block, a residual connection and a dropout layer of 10% are applied. When implementing the model, transfer learning is applied by loading the weights of the Kinetics-400 dataset (Kay *et al.*, 2017). Consequently, the final linear layer output feature is edited to represent the number of classes in the utilised dataset.

3.2.4 Video Dataset Sampling

To account for the varying length of videos, an effective and efficient video frame dataset loader was utilised[2]. A sampling strategy to represent the entire video was implemented as loading the entire sequence of frames is memory and compute intense (Wang *et al.*, 2016). Videos were uniformly sampled by splitting each video uniformly into 16 frames. Based on existing studies, the optimal number of video frames is 16 frames as fewer frames fail to represent the video and increasing the number of frames to 32 or 64 frames results in minimal accuracy gains while requiring significantly more computational demand (Sharir

[2]https://github.com/RaivoKoot/Video-Dataset-Loading-Pytorch

Table 3.1: Dataset folder structure.

```
Data
├── train
│   ├── train_annotations.txt
│   ├── absent # class folder naming
│   │   ├── signer0_sample170_color # video folder naming
│   │   │   ├── frame_00001.jpg
│   │   │   ├── ...
│   │   │   └── frame_00062.jpg
│   │   └── signer12_sample1114_color
│   │       ├── frame_00001.jpg
│   │       ├── ...
│   │       └── frame_00078.jpg
│   └── school
├── val
│   ├── val_annotations.txt
│   ├── absent
│   └── school
└── test
    ├── test_annotations.txt
    └── absent
```

et al., 2021, Tong *et al.*, 2022, Hrúz *et al.*, 2022). Both datasets must be structured accordingly to load the video frames into the models efficiently. Table 3.1 details the directory structure required for each dataset.

The dataset is split into the train, validation and test splits. The videos are subsequently split into their respective sign classes within each split. Each video contains a folder with the video frames renamed in the order they appear, starting at `img_00001.jpg` to `img_0000n.jpg`, where n is the last frame in the video. All RGB frames are saved as `.jpg` files, and pose landmarks are saved as `.npy` files. An annotations file is attached to each dataset split containing a list of the videos with the following information: `video_path`, `start_frame`, `end_frame`, `label_ID`.

3.2.5 Model Regularisation

The driving force behind signer-independent SLR is the ability to generalise to unknown signers. Therefore, constructing a robust model and applying essential generalisation methods are vital. Dropout and batch normalisation have already been implemented within the model layers in Section 3.2.1. Early stopping was implemented for all models, and data augmentation was applied to all vision-based models.

3.2.5.1 Data Augmentation

Data augmentation is applied directly to the data and not precomputed locally. The augmentation is implemented using the `transforms` library from the PyTorch framework. Augmentations are only applied to the training data, detailed in Listing 3.1. The only transformations applied to the validation and test data are the resizing and conversion to a tensor defined by the model's input shape. Geometric transformations are not applied to the pose landmark data to avoid distorting the spatial relationships between landmarks. For all vision-based models, the train, validation and test frames are resized to 256×256 except the InceptionV3-LSTM model, where frames are resized to 299×299. This is based on the input shape of the InceptionV3 model.

Data augmentation is employed through a `RandomApply` transformation with a 70% application probability to enhance model generalisability. Within this pipeline, two individual augmentations are defined: a `RandomHorizontalFlip` applied with a 50% chance and a `RandomAffine` transformation encompassing rotations, translations, scalings, and shears within specified ranges based on existing SLR studies (Pigou *et al.*, 2015, Rastgoo *et al.*, 2021a, Boháček and Hrúz, 2022). The horizontal flip is included to account for the signers being left- or right-hand dominant.

```
train_preprocess = transforms.Compose([
    ImglistToTensor(),
    transforms.Resize((image_size, image_size)),
    transforms.RandomApply(torch.nn.ModuleList([
        transforms.RandomHorizontalFlip(p=0.5),
        transforms.RandomAffine(degrees=(-7, 7), translate=(0.0, 0.33), scale=(0.8, 1.1),
    shear=(-7, 7)),
    ]), p=0.7)
])
```

Listing 3.1: Data augmentation pipeline.

3.2.5.2 Early Stopping

Early stopping is utilised during model training to prevent overfitting on the data. The validation loss metric is monitored with the aim of minimising it. The patience parameter is set to 25, giving the model 25 epochs to improve on the lowest validation loss before terminating training. The code for implementing an early stopping callback call with the `EarlyStopping` class is given in Listing A.5.

The initial lowest validation loss checkpoint and the next five consecutive checkpoints are saved. This is done to find the best-performing checkpoint on the test data. After training, all checkpoints are evaluated on the test split and the best checkpoint is chosen based on the highest accuracy.

3.2.6 Parameter Tuning

Model optimisation was implemented through parameter tuning. The parameters accessed for optimisation were learning rate and data augmentation ranges. Learning rates ranging from 1×10^{-3} to 5×10^{-5} were considered for from-scratch and transfer learning, respectively (Yim *et al.*, 2017).

Hyperparameter tuning for the ST-GCN model is applied to the temporal convolutional layer's kernel, k_t, to find the optimal kernel size for the given number of frames. The kernel size tuning range is from size one to nine for k_t, in increments of two.

The comparative analysis between a model trained with and without augmentation will provide insights into the impact of applying the data augmentation techniques detailed in Section 3.2.5.1 on enhancing the robustness and generalisation capabilities of the vision-based models for the construction of the complete system.

3.2.7 Training and Evaluation

Each model's training and evaluation pipeline incorporates the methodology approach discussed in Section 3.1.1 following Figure 3.2. Model parameters are based on the parameter tuning set out in Section 3.2.6, and all models utilised a batch size of 32.

3.2.7.1 Training

Before training, the loss function and optimiser have to be defined for the loss function `CrossEntropyLoss` is implemented, and `AdamW` is chosen as the optimiser with a weight decay coefficient of 1×10^{-2}, as seen in Listing 3.2. `AdamW` is considered better than the `Adam` optimiser for training a deep learning network because it incorporates weight decay directly into the optimisation process, as explained in Section 2.3.6.4. The `scheduler` controls the learning rate during training, decreasing the learning rate using a gamma of 0.5 after every 15 epochs.

```
1    # loss function
2    criterion = nn.CrossEntropyLoss()
3    # optimiser
4    optimiser = optim.AdamW(model.parameters(), lr=lr, weight_decay=1e-2)
5    # scheduler
6    scheduler = StepLR(optimiser, step_size=15, gamma=0.5)
```

Listing 3.2: Training loss function and optimiser.

The training pipeline in Listing A.6 configures the early stopping and iterates through data based on the number of epochs specified. The model weights are only updated when the effective batch size[3] is reached.

[3]The product of the batch size and accumulation factor. This technique is called gradient accumulation, which is utilised for larger batch sizes on limited GPU memory.

3.2.7.2 Evaluation

All models are evaluated on the validation set during training after each epoch, and training continues until epochs finish or early stopping is applied, as explained in Section 3.2.5.2. Once the training has concluded, the model is evaluated on the test split using the testing algorithm given in Listing A.7. As gradients are not being calculated and only model inferencing is performed, there is no need to update gradients. The ground truth and model predictions are accumulated to calculate the model accuracy and F-score performance explained in Section 4.1.1. This approach is used when evaluating the validation set during training and when testing the final model on the test set.

3.3 Summary

This chapter provided a high-level view of the overall system architecture, covering the various model architectures for evaluating signer-independent SLR. The dataset selection criteria with suitable datasets were motivated to evaluate signer-independent SLR successfully.

Furthermore, the implementation of the proposed system was explained, detailing the model architecture and regularisation techniques utilised. The MediaPipe pose estimation algorithm and landmark post-processing application were discussed to deal with failed landmark detections. The video data loading module covered the process of uniformly selecting the frames from the sign video clips. The parameter tuning criteria were covered to assess model performance based on the temporal kernel size and the effect of augmentations. Finally, the training and evaluation pipelines were presented to evaluate each model architecture.

4

Experimental Setup

This chapter provides details on the experimental setup used in this research, including the various metrics used to gauge model performance are examined in Section 4.1. The hardware and software of the system infrastructure are also provided in Section 4.3. Section 4.2 evaluates the signer-independent SLR datasets. Finally, the experiments conducted to examine the pose-based and vision-based approaches are described in Section 4.4.

4.1 Evaluating Classification Performance

Multiple metrics are available for gauging and comparing the performance of a given machine learning model. The most commonly used are accuracy, precision, recall, and F-score. For this respective study, accuracy and F-score are utilised to evaluate model performance and explained in the following subsections.

To comprehend the metrics, it is vital to distinguish between correctly and incorrectly predicted positive cases and correctly and incorrectly predicted negative cases. In this case, positive and negative refer to the prediction, and true or false denotes whether the prediction was correct or incorrect.

Table 4.1: Confusion Matrix Explanation.

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative	False Positive
	Positive	False Negative	True Positive

Table 4.1 can be defined as follows:

- True Positive (TP): A positive case was correctly predicted as positive.
- False Positive (FP): A negative case was incorrectly predicted as positive.
- True Negative (TN): A negative case was correctly predicted as negative.
- False Negative (FN): A positive case was incorrectly predicted as negative.

4.1.1 Accuracy

Classification accuracy is the more popular metric for evaluating and comparing the performance of models in the majority of isolated SLR research papers available (Huang *et al.*, 2015, Ronchetti *et al.*, 2016, Hara *et al.*, 2017, Al-qurishi *et al.*, 2021, Sincan and Keles, 2020, De Coster *et al.*, 2021).

Accuracy serves as a metric to gauge the overall correctness of a system (Hossin and Sulaiman, 2015). It is determined by computing the ratio of accurately classified samples to the total number of input samples, as given by Equation 4.1.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{4.1}$$

The accuracy metric to gauge model performance is contingent on a balanced dataset. This implies an even distribution of samples between the classes and signers within the dataset. If the dataset is imbalanced, the F-score metric, explained in Section 4.1.3, is preferred to measure a model's performance. The F-Score metric will, therefore, be reported alongside the accuracy score for imbalanced datasets.

4.1.2 Precision and Recall

Precision quantifies the proportion of correctly predicted positive instances out of all instances predicted as positive, given by Equation 4.2. Subsequently, defining the models'

reliability and providing a good measure to determine when the cost of false positives is high (Hossin and Sulaiman, 2015).

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

Recall is a valuable metric for assessing the classifier's comprehensiveness in identifying genuinely positive cases (Hossin and Sulaiman, 2015), given by Equation 4.3.

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

Precision and recall values are calculated per class, and the weighted average for the model can be computed by combining all the classes' precision and recall values. In contrast to accuracy, recall and precision are unaffected if the distribution of samples within the dataset is uneven owing to the weighted average calculation method, As mentioned in Section 4.1.1.

4.1.3 F-score

F-score is a combination of precision and recall scores, representing a harmonic mean between the two scores (Hossin and Sulaiman, 2015), calculated by Equation 4.4. The F-score metric provides a much more balanced measure to gauge a model's performance.

$$F\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.4}$$

In Chapter 5, the F-score is referred to as F1 for simplicity.

4.2 Datasets

Two SLR datasets have been selected for the examination of signer-independent SLR. The first dataset is a smaller Argentinian sign language, comprising 3200 videos, utilised for preliminary training and model experimentation. In conjunction, a more extensive signer-independent dataset, AUTSL, has been employed for the conclusive analysis of models and comparison with existing studies.

4.2.1 LSA64

The LSA64 dataset is an Argentinian publicly available dataset for isolated SLR research (Ronchetti *et al.*, 2016). The dataset contains video recordings of 64 sign gestures performed by ten non-expert signers, with both one- and two-handed sign gestures, as depicted in Figure 4.1. Each signer executes each sign five times, resulting in 3200 video samples. The right hand was used for all the one-handed signs, as seen in Figure 4.1a. In situations where single-handed signs are executed, the non-dominant left hand is consistently absent from the visual frame throughout the video clip. All signers wore black clothing and fluorescent gloves and performed signs centred against a white background. All subjects were non-signers and were subsequently taught to perform the signs during the recording session.

(a) Hungry. **(b)** Appear.

Figure 4.1: Snapshots extracted from the LSA64 dataset.

Naturally, the LSA64 dataset was not intended for signer-independent research. Therefore, to create a signer-independent version of the LSA64 dataset, the dataset was split

into three sections: training, validation, and test data. A 70:10:20 train:validation:test split was utilised. The signers and all their samples were isolated in each split such that no data leakage occurred across the splits. Signers 5 and 10 were isolated to the test set, Signer 7 to the validation set, and the rest to the train set.

Table 4.2: Breakdown of the LSA64 signs with the corresponding hand utilised to perform the sign, ordered by ID. R refers to the right hand, and B refers to both hands.

ID	Name	H	ID	Name	H	ID	Name	H
01	Opaque	R	23	Food	R	44	Rice	B
02	Red	R	24	Argentina	R	45	Barbecue	B
03	Green	R	25	Uruguay	R	46	Candy	R
04	Yellow	R	26	Country	R	47	Chewing-gum	R
05	Bright	R	27	Last name	R	48	Spaghetti	B
06	Light-blue	R	28	Where	R	49	Yogurt	B
07	Colors	R	29	Mock	B	50	Accept	B
08	Pink	R	30	Birthday	R	51	Thanks	B
09	Women	R	31	Breakfast	B	52	Shut down	R
10	Enemy	R	32	Photo	B	53	Appear	B
11	Son	R	33	Hungry	R	54	To land	B
12	Man	R	34	Map	B	55	Catch	B
13	Away	R	35	Coin	B	56	Help	B
14	Drawer	R	36	Music	B	57	Dance	B
15	Born	R	37	Ship	R	58	Bathe	B
16	Learn	R	38	None	R	59	Buy	R
17	Call	R	39	Name	R	60	Copy	B
18	Skimmer	R	40	Patience	R	61	Run	B
19	Bitter	R	41	Perfume	R	62	Realize	R
20	Sweet milk	R	42	Deaf	R	63	Give	B
21	Milk	R	43	Trap	B	64	Find	R
22	Water	R						

4.2.2 AUTSL

The AUTSL dataset is a Turkish large-scale multi-modal signer-independent SLR dataset (Sincan and Keles, 2020). The 38,336 sign video samples comprise 43 different signers performing 226 isolated sign gestures. The samples consist of various backgrounds recorded in indoor and outdoor venues, as seen in Figure 4.2. A complete visualisation of all the different backgrounds can be found in Appendix C.1.1. Along with the varying backgrounds,

the spatial positions and postures of the signers differ across the recordings. In recordings where signers utilise single-handed signs, the non-dominate hand is consistently visible throughout the video clip, resting in the signer's lap when seated or positioned by their side when standing. Table 4.3 gives a detailed breakdown of the statistics of the AUTSL dataset.

Figure 4.2: AUTSL sample video snapshots.

Table 4.3: AUTSL dataset statistics.

Property	**Description**
Number of Signs	226
Number of signers	43
Total samples	38,336
Number of different backgrounds	20
Mean samples per sign	169.6
Modalities	RGB, depth, skeleton
RGB and depth resolution	512×512
FPS	30

The number of samples for each signer ranges from 435 samples to 2697 samples, with an average number of samples per signer of 843, visualised in Figure 4.3. This highlights the imbalance across the signer sample distribution within the AUTSL dataset. The number of samples for each sign ranges between 90 and 127, with an average of 124. The signs *Sugar* and *Okay* are the two most underrepresented classes, with 90 and 95 samples each. The distribution of the number of samples of each sign within the train split and the hands used to perform each sign is also available to the interested reader in Appendix

C.1.2.

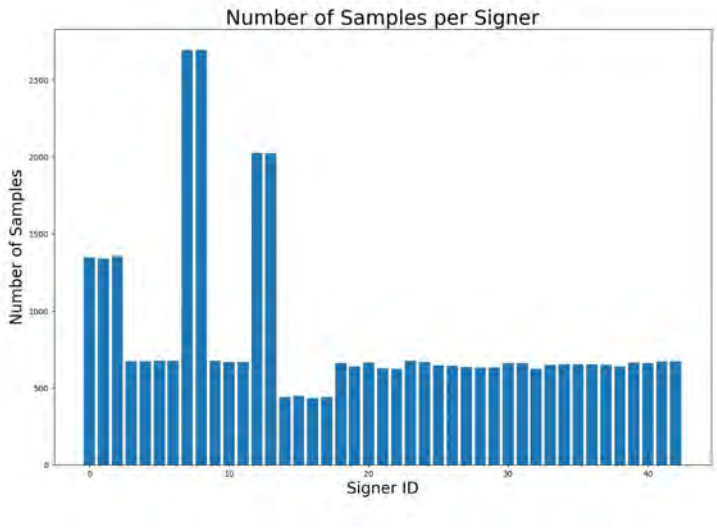


Figure 4.3: Distribution of samples per signer across the entire AUTSL dataset.

4.3 System Infrastructure

The building and testing the system's machine learning models required various hardware and software components as follows.

4.3.1 Hardware

A dedicated machine-learning server was created for use during this research. The machine learning server components included:

- CPU: AMD Ryzen™ 7 3800X 4.9 GHz 16-core
- RAM: G.Skill Ripjaws V 3600 MHz 64GB
- GPU: NVIDIA RTX 3090

4.3.2 Software

The following software was used to build, train and test the system:

- Ubuntu 22.04.3 Linux operating system
- Python 3.10.12
- PyTorch 2.0.1

4.4 Overview of Experiments

The following test model experiments were designed to satisfy the research objectives set out in Section 1.5.

4.4.1 Preliminary Experiments

Three feature extraction techniques were considered to determine the most appropriate approach for SLR: pose estimation, hand segmentation and raw RGB frames. The primary focus was on SLR manual features extracted from the hands. A 1D CNN was used to classify signs for the pose landmarks. Subsequently, a Pruned VGG and ResNet50V2 network were applied to the segmented and raw RGB frames. Although VGG-16 was initially considered, during preliminary experiments, pruning was necessary for the model to learn (Marais *et al.*, 2022). A signer-dependent version of the LSA64 dataset where the train, validation and test splits were randomly split, and signers were not isolated across splits using a 70:10:20 ratio was utilised for Experiment 1.1. Subsequent preliminary experiments focus on validating model training parameters as set out in Section 3.2.6. Therefore, the following preliminary experiments were implemented:

- Experiment 1.1 – Compares the performance of the three feature extraction methods on the smaller LSA64 dataset.

- Experiment 1.2 – Compares the temporal kernel size for the ST-GCN pose-based model on the LSA64 and AUTSL datasets.
- Experiment 1.3 – Compares the data augmentation parameter selection using the R(2+1)D-18 architecture on the AUTSL dataset.

4.4.2 Evaluation of the Pose-based Approach

To analyse the performance of the pose-based architecture on the two signer-independent datasets, the ST-GCN model was trained for the experiments listed below:

- Experiment 2.1 – Compares the performance of the ST-GCN architecture when trained with parameter tuning on the signer-independent LSA64 dataset.
- Experiment 2.2 – Compares the performance of the ST-GCN architecture when trained with parameter tuning on the AUTSL dataset.
- Experiment 2.3 – Compares the performance of the ST-GCN architecture when trained with zero-based imputation and using a combination of linear, zero-based and backwards fill imputation of missing frame landmarks.

The experiment results of the experiments are presented in Section 5.2.

4.4.3 Evaluation of Vision-based Approaches

Four variants of vision-based architectures, namely ResNet-LSTM, InceptionV3-LSTM, R(2+1)D-18 and Swin3D-T, were trained for the respective experiments listed below:

- Experiment 3.1 – Compares the performance of the architectures when trained on the signer-independent LSA64 dataset.

- Experiment 3.2 – Compares the performance of the architectures when trained on the AUTSL dataset.

Section 5.3 details the results of these experiments.

4.4.4 Evaluation of Transfer Learning

The effect of model performance based on utilising transfer learning is analysed on the best-performing vision-based model by the following experiment:

- Experiment 4.1 – Compares the performance of the top vision-based model when trained with and without transfer learning on the LSA64 dataset.
- Experiment 4.2 – Compares the performance of the top vision-based model when trained with and without transfer learning on the AUTSL dataset.

Section 5.4 presents the results of these experiments.

4.4.5 Number of Video Frames Selected

The number of frames selected from each video affects the performance of the various architectures. Therefore, the pose-based model was trained for the experiment listed below.

- Experiment 5.1 – Compares the performance of the ST-GCN model when trained and evaluated on different numbers of selected frames ranging from 4 to 32 on the AUTSL dataset.

The results of this final experiment are presented in Section 5.5.

4.5 Summary

This chapter detailed the accuracy and F-score metrics chosen to evaluate model performance for signer-independent SLR. A detailed breakdown of the two datasets was structured to train the models. The signer-independent AUTSL dataset was significantly more extensive and was therefore chosen as the final dataset for evaluating and comparing model performance. An overview of the eleven different experiments was provided, consisting of preliminary experiments, pose-based and vision-based architecture experiments, evaluation of transfer learning and analyses of the number of video frames selected. The next chapter reports on and analyses the results of those experiments.

5

Results and Discussion

This chapter presents and discusses the results of the tests conducted for the experiments detailed in Section 4.4.

5.1 Preliminary Experiments

The preliminary experiments examined the initial feature extraction approaches, parameter tuning of the ST-GCN temporal kernel size, and the effect of data augmentation for the vision-based approaches, set out in the experimental overview in Section 4.4.1.

5.1.1 Experiment 1.1 – Feature Extraction Methods

This experiment explores different spatial feature extraction techniques for SLR on the signer-dependent version of the LSA64 dataset. Therefore, pose estimation, hand segmentation and raw image input feature extraction techniques are analysed and compared without techniques from the temporal domain.

The hand pose-based model using the 1D-CNN architecture on the skeletal data achieved an accuracy of 94.91% on the unseen test data. The model was less adept at classifying the opaque and call sign classes, both single-handed signs.

The ResNet and Pruned VGG architectures architectures performed poorly on the segmented hands' data, with the ResNet model achieving the highest accuracy of 47.83%. The Pruned VGG model followed with an accuracy of 43.24%. For the raw image input,

the Pruned VGG marginally outperformed the ResNet architecture with an accuracy of 95.40%. The results of each model are given in Figure 5.1 along with a Conv3D model that incorporates the temporal domain, which achieved a near-perfect accuracy of 99.96%.

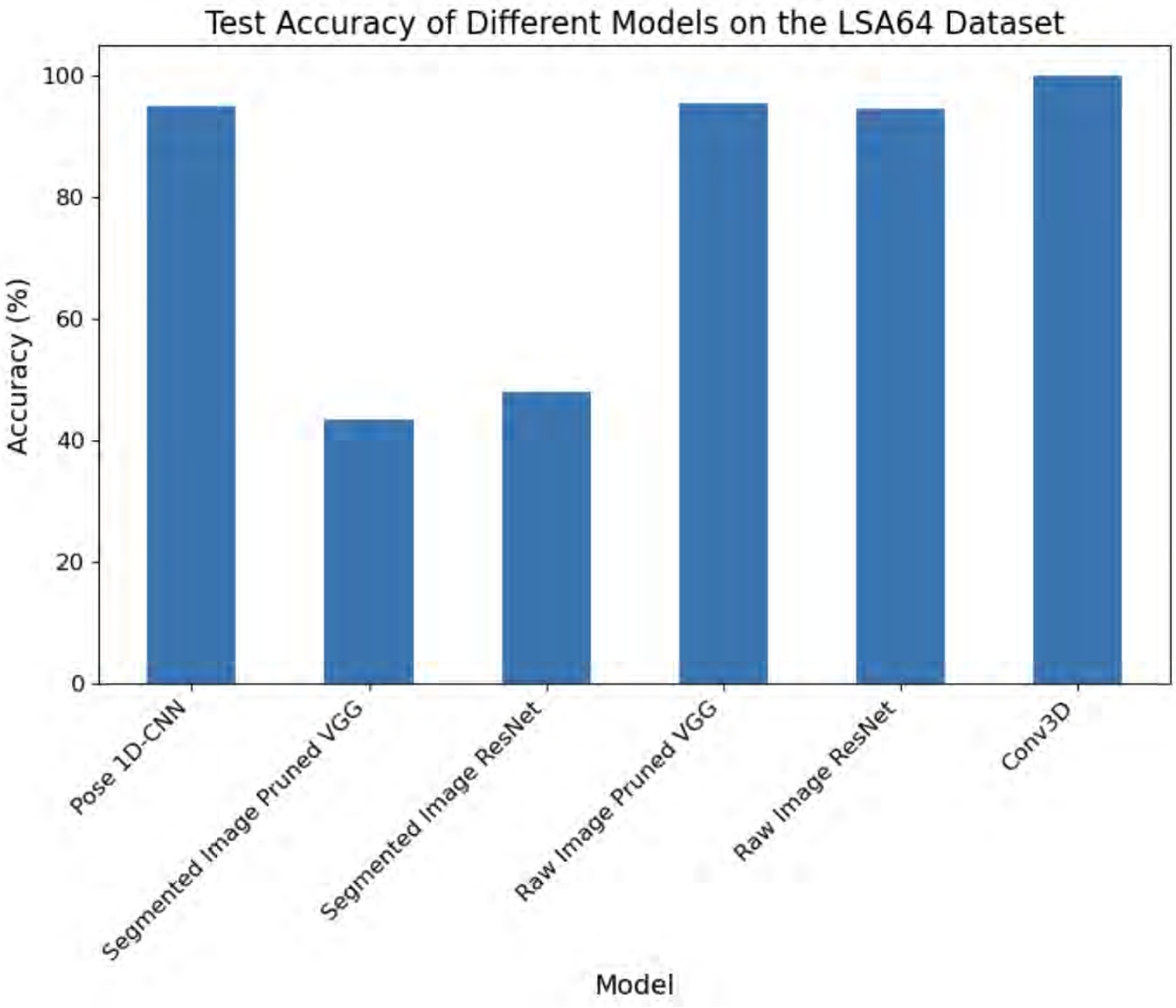


Figure 5.1: Experiment 1.1 – Accuracy performance of the architectures on the LSA64 preliminary dataset.

5.1.2 Experiment 1.2 – ST-GCN Temporal Kernel Size

The ST-GCN model is evaluated on various temporal kernel sizes, as explained in Section 3.2.6. The temporal kernel size, k_t, specifies the number of frame nodes analysed in each 1D convolution stride. A higher kernel size requires significantly more computational resources. Table 5.2 shows the various size kernels tuned on the LSA64 and AUTSL

datasets.

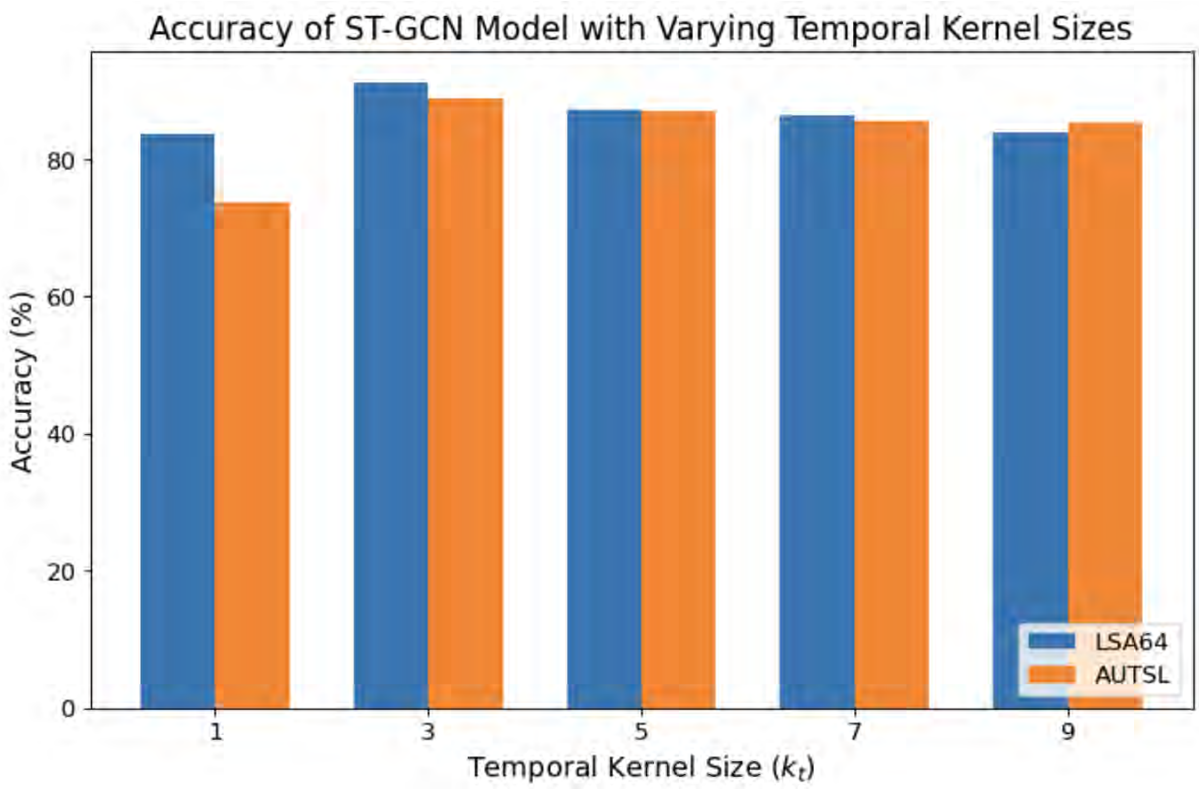


Figure 5.2: Experiment 1.2 – Hyperparameter tuning test accuracy of temporal size for the ST-GCN model on the LSA64 and AUTSL datasets.

The findings demonstrate that employing a smaller k_t size of three yields optimal performance, achieving accuracies of 91.25% and 88.95% on the test sets. As k_t size increases, model performance tends to decrease. The exception is k_t size of one, which fails to model temporal dependencies between video frames. The increase in k_t above 3 increases the number of frames being analysed in each stride. This may lead to more noise and the loss of finer temporal details between individual frames, thus decreasing the model's performance. Therefore, a temporal kernel of 3×1 is utilised in future experiments for the ST-GCN model with a stride of one.

5.1.3 Experiment 1.3 – Data Augmentation

Data augmentation validation is assessed to improve model generalisation ability and robustness. This is achieved by taking the R(2+1)D-18 architecture and training it with and

without the data augmentation parameters given in Section 3.2.5.1. The AUTSL dataset is utilised because it has significantly more variation across signers and backgrounds than the LSA64 dataset. Table 5.1 compares the results when trained with and without augmentation on the AUTSL dataset. A slight accuracy and F1 performance increase are noted with augmentations on the test split of the AUTSL dataset of 2.36% and 2.43%, respectively.

Table 5.1: Experiment 1.3 – R(2+1)D-18 Model Accuracy Performance with and without Augmentations.

Method	**Accuracy(%)**			**F1(%)**
	Train	**Validation**	**Test**	**Test**
With Augmentation	98.92	89.74	90.64	90.61
Without Augmentation	99.21	90.08	88.28	88.18

5.1.4 Discussion of Experiment 1

The raw image data using the Pruned VGG architecture achieved the best accuracy performance out of the three preliminary experiments. The Pruned VGG and ResNet encountered difficulties with the segmented hand's data due to the lack of spatial and temporal information (Garcia-Garcia *et al.*, 2018). While the segmentation of the hands was nearly perfect, as illustrated in Figure 5.3, the constraining factor arose from the insufficient spatial awareness regarding the body's location in the image.

(a) Appear sign snapshot. **(b)** Segmentation of both hands from the sign.

Figure 5.3: Segmentation of both hands on the Appear sign from the LSA64 dataset.

MediaPipe Holistic provides a computationally efficient method of extracting hand landmarks and training a 1D-CNN for SLR. Given limited computational resources, hand

landmark data combined with a 1D-CNN for isolated SLR is feasible. However, if computational resources are not limited, vision-based approaches such as the Pruned VGG or ResNet are good alternatives with minimal preprocessing required. The RGB vision-based and pose-based landmarks approaches align with the ideal real-world scenario of an SLR system, not requiring specialised equipment or visual markers. The Conv3D architecture showed an improvement of 4.56% on the top-performing Pruned VGG architecture, highlighting the importance of including temporal information in the SLR system.

The optimal temporal depth kernel for the ST-GCN was $(3, 1)$, as it was found there was a negative correlation between the kernel size and model performance on both the LSA64 and AUTSL datasets decreased. Consequently, this improves the model's computational intensity as a smaller kernel requires significantly less computational processing power. Moreover, the smaller kernel size benefits isolated SLR video clips by more effectively capturing short-term temporal dependencies, providing improved temporal resolution to distinguish between closely spaced sign events.

Experiment 1.3 showed that using augmentations slightly increases accuracy for the vision-based R(2+1)D-18 model. The lack of substantial improvement with augmentations exemplifies the extensive variation within the AUTSL dataset splits. Furthermore, the intra-class variation in signer start and end positions adds an additional aspect to consider, with some signers sitting and others standing and not always centred in the video. This can reduce overall accuracy since the system's ability to discriminate between different signs can be confused with a different signer. Consequently, increasing the inter-class variation which is an undesired effect.

5.2 Evaluation of the Pose-based Approach

This section presents and discusses the results and analysis of the ST-GCN pose-based approach for signer-independent SLR. An ablation study is also conducted in Experiment 2.3 to assess the impact of frame interpolation and imputation.

5.2.1 Experiment 2.1 – ST-GCN on LSA64

The ST-GCN model was evaluated on the LSA64 dataset. As indicated by the parameter tuning results in Section 5.1.2, a 3×1 temporal kernel was applied. Figure 5.4 visualises the training and validation accuracy curve for the ST-GCN Model on the LSA64 dataset. The ST-GCN model yielded accuracies of 99.73% and 93.75% on the train and validation sets, respectively. Early stopping was applied at epoch 98 to avoid overfitting. On the test set, the model achieved an accuracy of 91.25% and F1 of 91.06%.

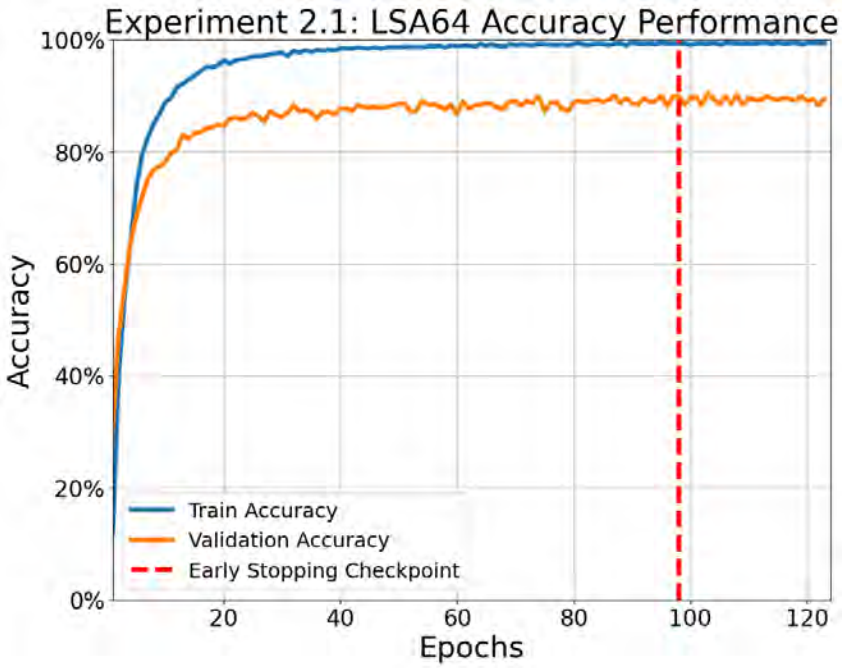


Figure 5.4: Experiment 2.1 – Training and validation accuracy performance of the ST-GCN pose-based approach on the LSA64 dataset.

The ST-GCN model encountered difficulties classifying specific isolated sign classes, namely *Bright*, *Yogurt*, *Drawer* and *Born*, within the LSA64 dataset. This is evident in the individual one-vs-rest F1 percentages, which ranged from 46.15% to 62.50%. The complete classification report can be found in Appendix B.1. Further analysis of the classes revealed that the model often incorrectly misclassified the *Realize* class as the *Buy* class. The confusion is apparent in Figure 5.5 as both signs are performed with a single hand and within a similar visual-spatial region and representation. For the interested reader, the confusion matrices for Experiments 2 and 3 can be found in Appendix B.5.

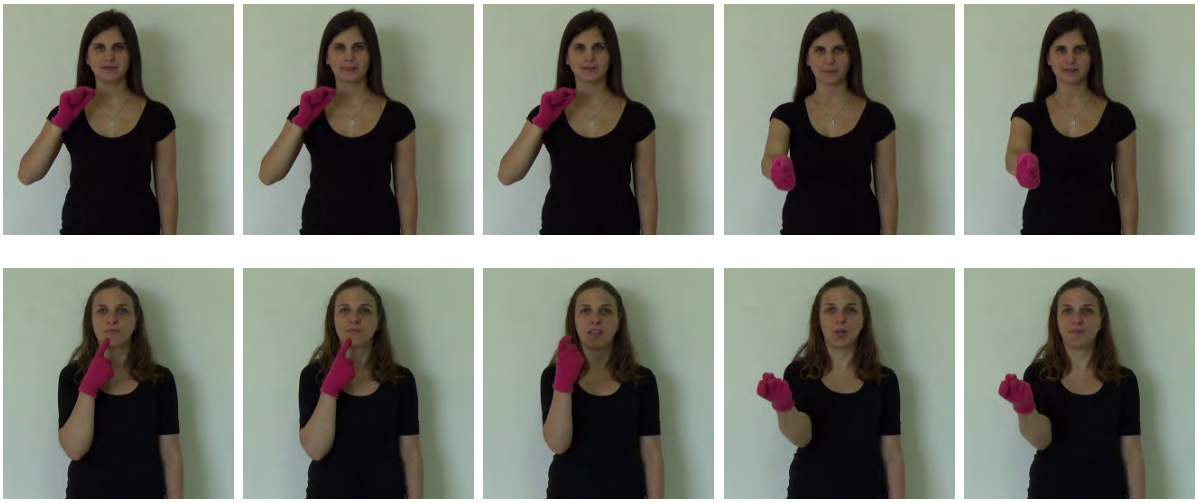


Figure 5.5: Experiment 2.1 – ST-GCN confusion between the predicted *Buy* class (top row) and actual *Realize* class (bottom row). For comparison, five sequential frames were selected at even intervals across the entire video clip from the LSA64 test split.

5.2.2 Experiment 2.2 – ST-GCN on AUTSL

The performance of the ST-GCN architecture was thoroughly evaluated on the AUTSL dataset. As with Experiment 2.1, a temporal kernel of size $(3, 1)$ was applied. The larger, more complex AUTSL dataset was utilised to gauge the model's generalisation performance. The ST-GCN model trained over 98 epochs achieved training and validation accuracies of 99.35% and 89.94%, respectively, as seen in Figure 5.6. A test accuracy of 88.95% was yielded on the test set with a corresponding F1 of 88.76%.

Comparing the model performance across sign classes, the ST-GCN model experienced difficulties with the *School*, *Tailor* and *Government* classes, with one-vs-rest F1 percentages of 46.67%, 47.62% and 54.55%, respectively. A further breakdown of the classification report can be found in Appendix B.2. Detailed analysis of the predicted versus actual class revealed that the *Government class* was frequently misclassified as the *Atatürk class*. Visually, the signs are similar, with the slight difference being that the index finger in the *Government* class points towards the mouth, whereas the index finger points towards the eye for the *Atatürk* class, as depicted by Figure 5.7. Contextually, both signs have origins in the reformed Turkish language when Turkey was declared a republic by the 1st President Atatürk (Tachau, 1964).

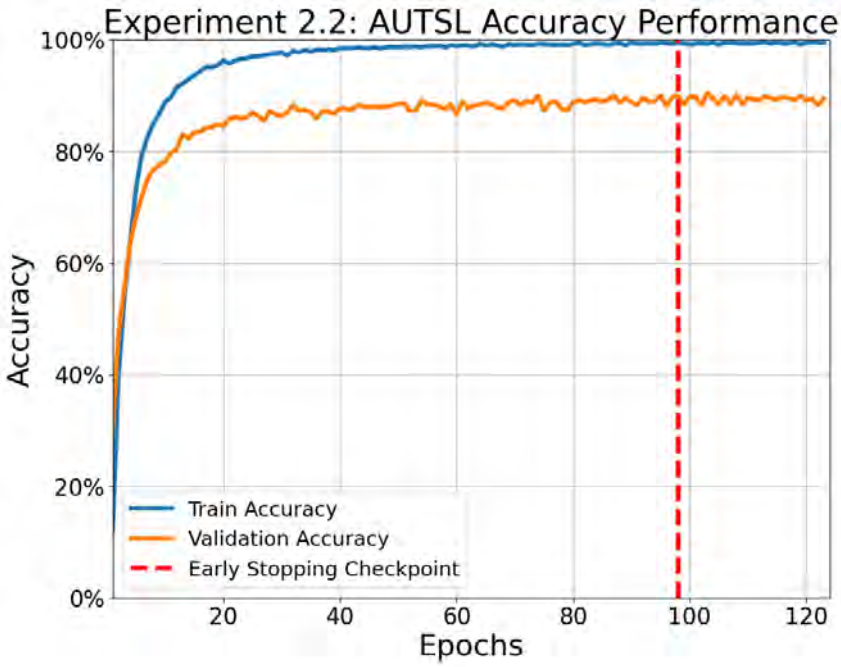


Figure 5.6: Experiment 2.2 – Training and validation accuracy performance of the ST-GCN pose-based approach on the AUTSL dataset.

5.2.3 Experiment 2.3 – ST-GCN Frame Interpolation

Frame interpolation and imputation are vital to deal with missing landmarks. Therefore, Experiment 2.3 validates the impact of post-processing of the pose estimation landmarks, discussed in Section 3.2.2.2 by examining using zero-based imputation on missed frames compared to performing linear imputation, backwards-filling and, as a last resort, zero-based imputation. Zero-based imputation is only utilised if the landmark subset for the hand or body is absent from the entire video clip.

Table 5.2 shows that utilising the full post-processing imputation on the LSA64 dataset, there is a 5.00% accuracy increase in model performance. On the AUTSL dataset, there is a 2.54% accuracy gain when applying the full post-processing imputation technique compared to just utilising zero-based imputation for missed landmarks.

Figure 5.7: Experiment 2.2 – ST-GCN confusion between the predicted *Atatürk* class (top row) and actual *Government* class (bottom row). For comparison, five sequential frames were selected at even intervals across the entire video clip from the AUTSL test split.

Table 5.2: Experiment 2.3 – Comparison of the post-processing imputation of landmarks on the LSA64 and AUTSL dataset using the ST-GCN model.

Test Setup	**Accuracy(%)**			**F1(%)**
	Train	**Validation**	**Test**	**Test**
LSA64 Zero-based Imputation	99.38	92.19	86.25	85.46
LSA64 Full Imputation	99.73	93.75	91.25	91.06
AUTSL Zero-based Imputation	97.63	88.53	86.41	86.28
AUTSL Full Imputation	99.35	89.94	88.95	88.76

5.2.4 Discussion of Experiment 2

The ST-GCN pose-based approach for signer-independent SLR showed promise by constructing a spatiotemporal graph and mapping human body landmarks with an adjacency matrix, as discussed in Section 3.2.2.3. When comparing the model performance between the LSA64 and AUTSL datasets, the model seems to struggle with the much simpler LSA64 dataset when considering the notable difference between the two datasets. This can be attributed to the high number of missing left-hand landmarks, as discussed in Section 3.2.2.2, within the LSA64 dataset. If the left hand is not utilised in the sign being performed in the LSA64 dataset, it is completely removed from the video clip. In comparison, for the AUTSL dataset, if only one hand is used in the performed sign, the other hand is kept stationary either on the signer's lap or side, always visible in the video

clip.

Figures 5.4 and 5.6 illustrate an overfitting problem. This is expected due to the nature of the problem being addressed of signer-independent SLR and the significant disparity between the train and validation sets. Notably, the model took the same number of epochs to train on both datasets, showcasing the ST-GCN's efficiency in extracting features and scaling to larger datasets. The landmark post-processing technique discussed in Section 3.2.2.2 led to an improvement in model performance on both datasets, especially with the LSA64 dataset due to the much higher number of missed landmarks that were interpolated, which can be seen in Figure 3.4.

Comparing the pose-based approach to existing studies on the AUTSL datasets, the proposed ST-GCN model performs well as a single-modality approach for signer-independent SLR. The ST-GCN model surpassed the findings of Moryossef *et al.* (2021)'s study, discussed in Section 2.2.1, where an accuracy of 85.63% was achieved by integrating MediaPipe holistic with a transformer architecture employing 75 landmarks. Compared to the SPOTER architecture discussed in Section 2.2, which employed MMPose, the ST-GCN model achieved 4.05% better performance on the AUTSL dataset.

On the other hand, the proposed ST-GCN model is outperformed by the single-modality SL-GCN network from SAM-SLR in Jiang *et al.* (2021)'s study, achieving an accuracy of 95.45% on the AUTSL dataset. The increased performance of the SL-GCN block is likely due to the reduced number of nodes, 27, which simplifies the problem being solved and may affect the representation of the human skeletal data. Notably, the ST-GCN block incorporates spatial and temporal information in the model. In contrast, the SL-GCN block primarily focuses on spatial information on single-view images. This could have led to the model's better performance when combined with the SSTCN architecture, which introduced the temporal component.

5.3 Evaluation of Vision-based Approaches

The performance results of the vision-based architectures, namely the ResNet50-LSTM, InceptionV3-LSTM, R(2+1)D-18 and Swin3D-T, are presented in this Section. The vision-based models are performed following the implementation discussed in Section 3.2.

5.3.1 Experiment 3.1 – Vision-based Approaches on LSA64

The performance of the vision-based models is analysed on the smaller, less complex LSA64 dataset. Based on the preliminary Experiment 1.3 in Section 5.1.3, the data augmentations discussed in Section 3.2.5.1 were applied when training each model. All models achieved accuracies and F1s above 90.00% on the test split, as described in Table 5.8. Furthermore, the R(2+1)D-18 model achieved the best performance on the LSA64 dataset, yielding an accuracy and F1 of 99.69% and 99.69%, respectively. Subsequently, it was followed closely by the Swin3D-T model with near-perfect accuracy and F1.

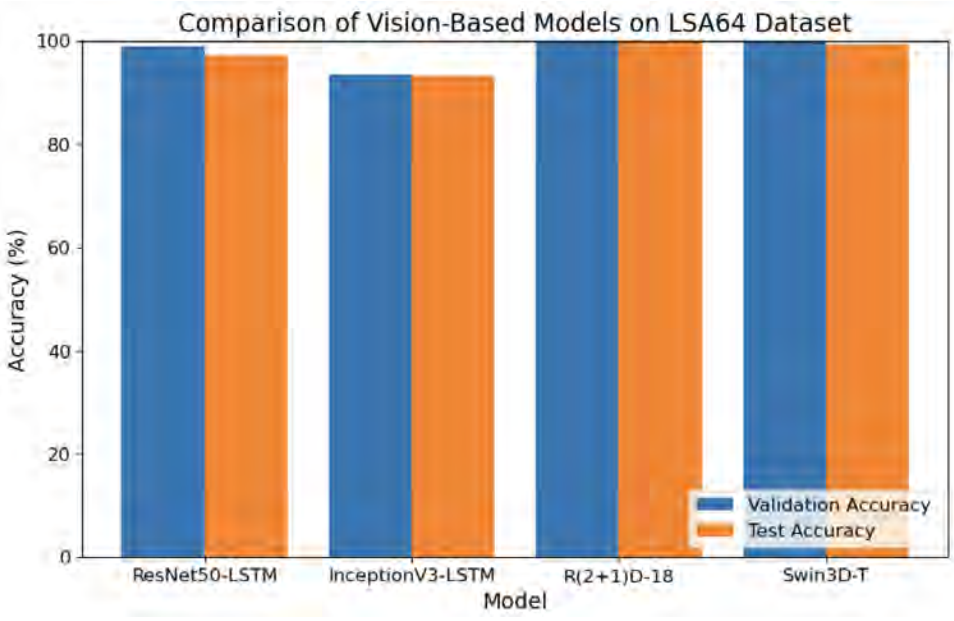


Figure 5.8: Experiment 3.1 – Comparison of vision-based models performance on the LSA64 dataset.

The two baseline models performed strongly on the smaller dataset, with the InceptionV3-LSTM model slightly overfitting on the train set of the LSA64 dataset. Specifically, the

R(2+1)D-18 model performance dropped with the individual *Chewing gum*, *Water*, *Call* and *Deaf* classes. The model achieved one-vs-rest F1 percentages of 94.74% for the initial two classes and 95.25% for the subsequent two classes. All four of these signs are single-hand signs. For the other classes, the model recorded a perfect F1 of 100.0%. Analysing the confusion matrix for the R(2+1)D-18 model, the *Water* and *Deaf* classes were misclassified as *Call* and *Uruguay* signs, respectively.

5.3.2 Experiment 3.2 – Vision-based Approaches on AUTSL

The vision-based models' performance and generalisation ability are evaluated on the larger, more complex AUTSL dataset for signer-independent SLR. The same data augmentation setup explained in Experiment 3.1 is utilised in Experiment 3.2 with identical data augmentation parameters. Examining Table 5.9, the hybrid CNN-RNN architectures faced difficulties in effectively addressing the complexities in the AUTSL datasets. The hybrid architectures overfitted on the train set and yielded accuracies of 71.13% and 67.98% for the ResNet50-LSTM and InceptionV3-LSTM models, respectively, on the test set. However, when utilising the R(2+1)D-18 and Swin3D-T architectures, model performance improved, achieving accuracies of 90.64% and 87.93%, respectively.

The best-performing vision-based model on the AUTSL dataset was the R(2+1)D-18, which yielded an accuracy of 90.64% and an F1 of 90.61%. Across the two top-performing models, both architectures experienced difficulties with certain sign classes from the AUTSL dataset, which were the *Lightweight* and *Fasting* signs, with F1 ranging between 42% and 56%. A complete sign class classification report from the R(2+1)D-18 model can be found in Appendix B.4. The R(2+1)D-18 architecture often incorrectly classified the *School* sign as *Soup* and the *Police* sign as *Rent* in the AUTSL test set, which can be seen in Figure 5.10 and 5.11, respectively.

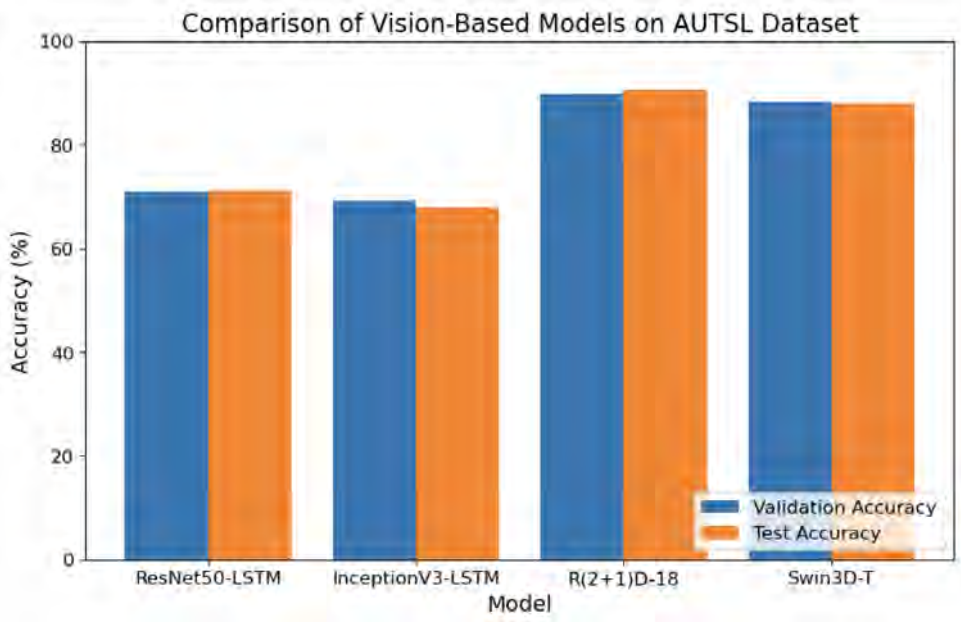


Figure 5.9: Experiment 3.2 – Comparison of vision-based models performance on the AUTSL dataset.

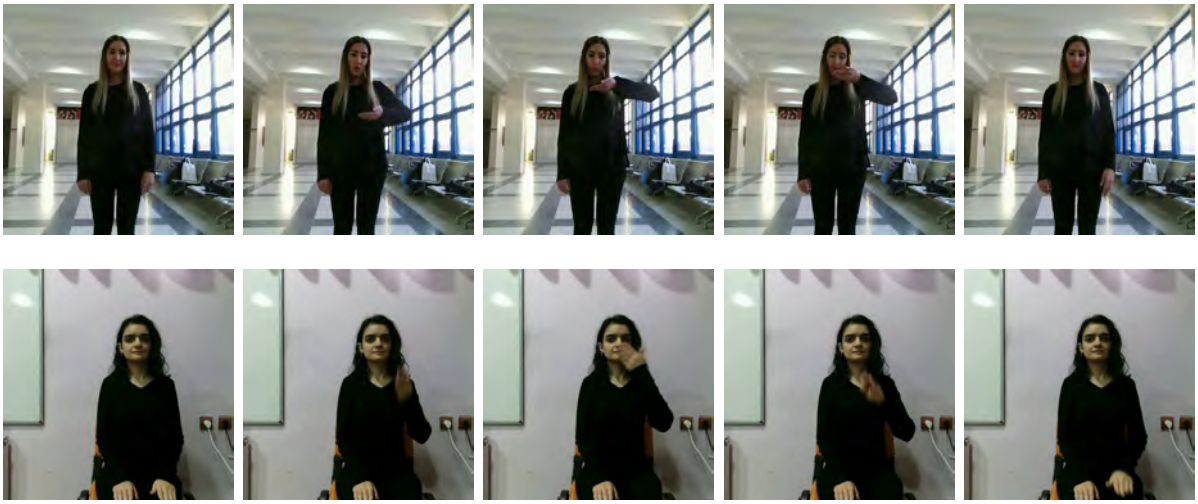


Figure 5.10: Experiment 3.2 – R(2+1)D-18 confusion between the predicted *Soup* class (top row) and actual *School* class (bottom row). For comparison, five sequential frames were selected at even intervals across the entire video clip from the AUTSL test split.

Figure 5.11: Experiment 3.2 – R(2+1)D-18 confusion between the predicted *Police* class (top row) and actual *Rent* class (bottom row). For comparison, five sequential frames were selected at even intervals across the entire video clip from the AUTSL test split.

5.3.3 Discussion of Experiment 3

Generally, the vision-based architectures for signer-independent SLR are robust single-modality performers on the LSA64 and AUTSL datasets. The baseline CNN-RNN architectures performed well on the LSA validation and test sets but failed to generalise well when presented with the AUTSL validation and test sets. The overfitting in the hybrid CNN-RNN could be attributed to model complexity, with the hybrid models having a higher complexity than the R(2+1)-18 and Swin3D-T architectures. On the contrary, the spatiotemporal convolution and vision transformer architectures performed well on both datasets. On the AUTSL test set, accuracies of 90.64% and 87.93% were achieved by the spatiotemporal convolution and vision transformer, respectively.

It is important to note that even with data augmentation and model regularisation, as discussed in Section 3.2.5, the vision-based architectures still overfit the training set of the AUTSL dataset. Therefore, it highlights again the complexity between the dataset splits of the AUTSL dataset and training data sufficiency.

The R(2+1)D-18 spatiotemporal convolutional network demonstrated superior performance in the signer-independent SLR action recognition task compared to the newer Swin3D-T video vision transformer approach. This emphasises the continued effective-

ness of CNNs in achieving better results for action recognition tasks. However, the video vision transformer is still relatively new to computer vision and video understanding and on that account, it may improve in the future. As the study focused on single isolated sign video clips, when presented with videos containing multiple signs or entire sentences, the vision transformer architecture could handle the long-term dependencies better across signs for continuous SLR.

Comparing the proposed vision-based approaches to existing studies on the AUTSL dataset using only RGB input data, all models outperformed the baseline architecture, CNN+FPM+BLSTM+Attention, implemented by Sincan and Keles (2020), as seen in Table 5.3. Analysing the single-modality approaches, both the Swin3D-T and R(2+1)D-18 models outperformed the TimeSformer approach and were only outperformed by the unimodal ResNet50-I3D architecture. As expected, the RGB multi-modality or ensemble-based architectures had the most robust performance on the AUTSL dataset, with the SAM-SLR architecture achieving the highest accuracy of 96.96%.

Table 5.3: Model accuracy performance compared to existing studies on the AUTSL test dataset using RGB data.

Architecture	**Accuracy(%)**
CNN+FPM+BLSTM+Attention (Sincan and Keles, 2020)	47.62
InceptionV3-LSTM	**67.98**
ResNet50-LSTM	**71.13**
TimeSformer (Hrúz *et al.*, 2022)	85.89
Swin3D-T	**87.93**
R(2+1)D-18	**90.64**
ResNet50-I3D (Hrúz *et al.*, 2022)	92.86
VTN+Pose flow (De Coster *et al.*, 2021)	92.92
Neural Ensembler (Hrúz *et al.*, 2022)	96.37
SAM-SLR (Jiang *et al.*, 2021)	96.96

In summary, the vision-based models demonstrated notable improvements over the baseline architectures. The proposed models performed competitively with state-of-the-art models on the AUTSL dataset, showcasing their efficacy in unimodal signer-independent SLR tasks. Furthermore, the R(2+1)D-18 architecture was the top-performing vision-based model from the models implemented.

5.4 Evaluation of Transfer Learning

Transfer learning leverages pretrained models on large datasets, significantly improving model performance. The following experiments compare the best vision-based model from Experiment 3.2 when trained with and without the pretrained weights from the Kinetics-400 dataset.

5.4.1 Experiment 4.1 – Transfer Learning on LSA64

The effect of transfer learning using the Kinetics-400 dataset on the R(2+1)D-18 architecture for the LSA64 dataset is evaluated. The initial model trains the R(2+1)D-18 model on the LSA64 dataset from scratch without any pretrained weights. The subsequent R(2+1)D-18 model loads the pretrained weights from the Kinetics-400 dataset before training on the LSA64 dataset. Table 5.4 compares the model's performance when training from scratch and utilising the Kinetics-400 pretrained model. Analysing the results, transfer learning benefits the model's overall performance with an increase of 4.22% in accuracy on the test set with a near-perfect accuracy of 99.69%.

Table 5.4: Experiment 4.1 – R(2+1)D-18 Model Accuracy Performance with and without Transfer Learning from the Kinetics-400 dataset on the LSA64 dataset.

Method	**Accuracy(%)**			**F1(%)**
	Train	**Validation**	**Test**	**Test**
From scratch	99.20	94.37	95.47	95.17
With Transfer Learning	100.0	100.0	99.69	99.69

When trained from scratch, the model took 89 epochs to implement early stopping, and when trained with transfer learning, early stopping was applied after 102 epochs.

5.4.2 Experiment 4.2 – Transfer Learning on AUTSL

This experiment builds on the previous experiment by examining the effect of transfer learning on the larger high-variation AUTSL dataset using the R(2+1)D model. The

model was chosen based on the best performance among the vision-based models in Experiment 3.2 on both datasets. As with Experiment 4.1, the model is trained from scratch and with the pretrained Kinetics-400 dataset weights. There is a significant improvement in the train and validation loss performance of the model when utilising the pretrained model. The final validation loss for the best epoch for the model trained from scratch was 1.01 compared to the validation loss of 0.55 for the pretrained model, as evidenced by Figure 5.12. Consequently, this yields a higher accuracy of 90.64% on the AUTSL test compared to 75.64% when trained from scratch.

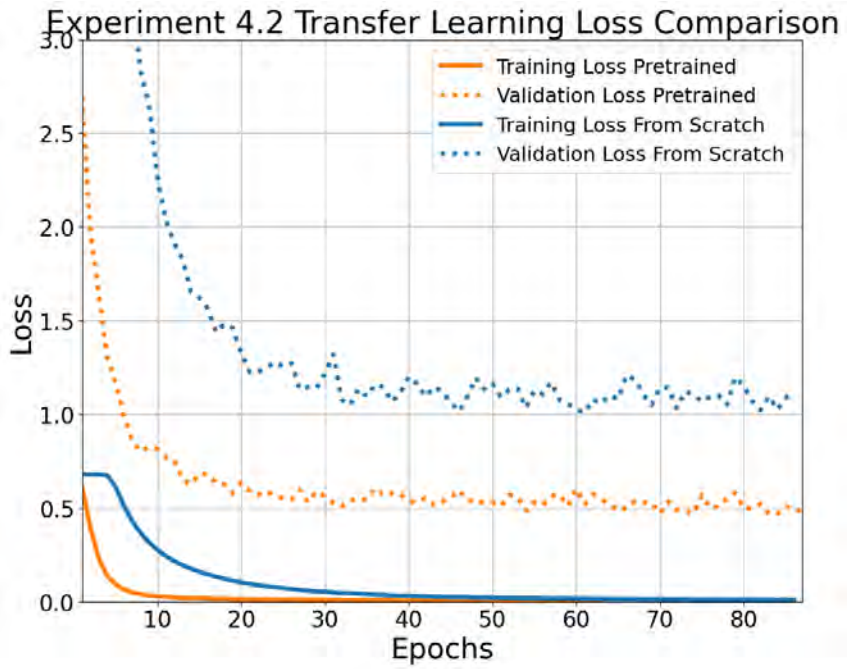


Figure 5.12: Experiment 4.2 – R(2+1)D-18 model accuracy performance when trained from scratch and with transfer learning from the Kinetics-400 dataset on the AUTSL dataset

Analysing the training time, the pretrained model took 81 epochs to implement early stopping and complete training, whereas the model trained from scratch took 86 epochs.

5.4.3 Discussion of Experiment 4

Using a pretrained model led to a substantial decrease in both train and validation loss performance on the larger AUTSL dataset, with the pretrained model achieving a signifi-

cantly lower final validation loss of 0.55 compared to the model trained from scratch with a validation loss of 1.01. Corresponding to a notable validation accuracy improvement from 75.61% to 89.74% when using pretrained weights.

The training time on the larger AUTSL dataset was shorter when utilising the pretrained model. This is primarily because transfer learning offers better starting weights when learning general features from a benchmark dataset, making it more efficient in adapting to a high-variation dataset such as AUTSL. In contrast, the longer training time on the smaller LSA64 dataset results from the pretrained features from the Kinetics-400 dataset not being as relevant to capture the dataset-specific patterns[1].

To further harness transfer learning, an effective strategy involves fine-tuning the model while selectively freezing specific layers, as seen with Jiang *et al.* (2021)'s study in Section 2.2 – increasing the overall accuracy of the model by 0.91%. However, the additional time and resources required to train the model are substantial and beyond the scope of this research.

5.5 Number of Video Frames Selected

A crucial part of working with video clips and temporal data is deciding on the optimal number of frames selected from each video that best represents the performed sign. Therefore, Experiment 5.1 explores different frame ranges on the AUTSL dataset using the pose-based ST-GCN model. Compared to the vision-based approaches, the ST-GCN model is utilised owing to its significantly faster training time and reduced computational resource requirements.

[1]The step down in high-variation Kinetics-400 and low-variation LSA64 datasets hinders the transfer learning process as opposed to the high-variation AUTSL.

5.5.1 Experiment 5.1 – ST-GCN Optimal Frame Selection

An analysis of how the number of frames uniformly selected from each sign video affects model performance on the AUTSL dataset is conducted. The frame values considered were 4, 8, 16 and 32. The length of the shortest video limited the number of frames selected to 32 and computational time resources.

Table 5.5: Experiment 5.1 – Comparison of performance of ST-GCN model based on the number of video frames selected.

Frames	**Accuracy(%)**			**F1(%)**
	Train	**Validation**	**Test**	**Test**
4	92.93	82.86	78.57	78.32
8	97.66	88.12	84.40	84.38
16	99.35	89.94	88.95	88.76
32	99.06	90.28	88.28	88.24

From the experiment, the optimal number of frames was found to be 16 frames, with the best test accuracy and F1 of 88.95% and 88.76%, respectively. This was closely followed by 32 frames with an accuracy of 88.28%, as seen in Table 5.5. Selecting 4 and 8 frames significantly dropped model performance, failing to capture the entire video clip information. It is also important to note the 35.69% increased time to train 32 frames compared to 16 frames, with 32 frames taking 7h 3m 22s and 16 frames taking 4h 32m 21s to train.

5.6 Summary

This chapter presented and discussed the results of all the experiments conducted for pose-based and vision-based approaches towards isolated signer-independent SLR. Furthermore, parameter validation and preliminary experiments were utilised to improve model performance and guide subsequent experiments.

The preliminary experiments showed that the best input data source was either using pose landmarks or the raw RGB data, as segmenting the hands resulted in a significant performance drop.

Analysing the results, it was found that the R(2+1)D-18 model was the top-performing model on the LSA64 and AUTSL datasets, with a final accuracy of 90.64% on the AUTSL dataset. The pose-based ST-GCN model closely followed the vision-based model. The notable difference between the vision-based approach and the pose-based was the significantly faster training time and reduced computation demand of the latter.

The ablation study analysed the effect of post-processing the pose landmarks using interpolation and imputation. It demonstrated the importance of correctly handling missing frame landmark data, resulting in a significant accuracy gain of 2.54% on the AUTSL dataset. Moreover, it was found that the selection of 16 frames from each video clip for training was the optimal frame selection number based on performance and computational demand.

6

Conclusion and Future Work

This chapter concludes the thesis, emphasising the contributions made towards the research in SLR and outlining potential avenues for future work.

6.1 Conclusion

This research used unimodal pose-based and vision-based approaches towards isolated signer-independent SLR using RGB video input on the LSA64 and AUTSL datasets. The unimodal RGB-only input approach is based on the real-world setting of SLR, where other data sources are not readily available or require specialised equipment.

Suitable publicly available datasets were selected for the evaluation and comparability of the proposed approaches. The LSA64 and AUTSL datasets were thus selected. While the LSA64 dataset is a smaller dataset with less variation, consisting of 64 signs for initial experimentation and exploration, the AUTSL dataset is a significantly larger dataset consisting of 226 and a substantial amount of signer and background variation formulated to evaluate models on the signer-independence problem.

Utilising the RGB input, two approaches were proposed: a pose-based approach and a vision-based approach. The pose-based approach applied the MediaPipe Holistic pose estimation algorithm to extract landmarks from the video frames. To overcome the complication of the failed frame detections by the pose estimation algorithm, frame interpolation and imputation were implemented on the effect videos. Subsequently, to perform training on the pose data, an ST-GCN architecture was proposed to model the spatial and temporal information. The vision-based approach analysed various CNN architectures and

a Vision Transformer model. For the CNN architectures, two hybrid CNN-RNN models were proposed using ResNet50 and InceptionV3 for the feature extractors for the spatial domain and an LSTM to classify and incorporate temporal data. Spatiotemporal convolutions were employed for the final CNN model, an R(2+1)D-18 model using ResNet18 architecture combined with a 1D CNN to model temporal data. The tiny version of the Video Swin Transformer, Swin3D-T, was implemented to study the effectiveness of transformers for signer-independent SLR.

The final system was constructed using the following steps: model builder, feature extraction, model training and model evaluation. Both datasets were divided into isolated training, validation, and testing data splits and loaded using a video dataset loader, which uniformly selected 16 frames from each video. Augmentation was applied to the training data for the vision-based approach. Finally, each model was tested on the unseen test data to evaluate and compare model performance.

Overall, 11 test scenarios were conducted to analyse isolated signer-independent SLR on both datasets, primarily focusing on evaluating model performance on the AUTSL dataset. The R(2+1)D-18 vision-based model was the top-performing model with an accuracy of 90.64% on the unseen AUTSL dataset test split. The pose-based ST-GCN model closely followed this with an accuracy of 89.95%. Additionally, the pose-based approach generalised well to the substantial background and signer variation in the AUTSL dataset by abstracting each signer to a set of landmarks without requiring data augmentation on the training set. However, it is essential to note that this relies on the performance of the pose estimation algorithm and landmark post-processing techniques to deal with failed detections. The main strong point of the pose-based approach is the significantly less computational power required and reduced training time compared to vision-based approaches.

Compared to existing studies, the top-performing proposed pose-based and vision-based approaches outperformed existing unimodal approaches, such as the SPOTER pose-based approach and TimeSformer vision-based architecture on the AUTSL dataset. Generally, the proposed approaches were only outperformed by multimodal architectures utilising a

combination of pose and vision data and RGB-depth data to train the model.

In conclusion, the overall findings of this research showed that the proposed pose-based and vision-based systems effectively classified sign classes in the LSA64 and AUTSL datasets. Although the results on the AUTSL dataset are promising, it is still clear that multimodal approaches tend to outperform unimodal systems. However, it is significant to note that in a real-world setting, limited data source availability and the trade-off of accuracy and computational resources when comparing unimodal and multimodal systems. Further accuracy improvement is possible through the potential use of fine-tuning using the validation set. Alternatively, a multimodal approach that only utilises RGB input but incorporates both vision and pose data to perform classification is also possible.

6.2 Contributions

This study contributed significantly to the SLR field, subsequently achieved through the initial research questions and objectives.

6.2.1 Research Questions

This study has provided answers to the research questions set out:

1. Do pose-based algorithms improve signer-independent SLR performance over vision-based algorithms? In contrast, experiments show that the vision-based architectures slightly outperform pose-based approaches with both approaches being state-of-the-art unimodal architectures for signer-independent SLR.

2. Are pose-based algorithms more computationally efficient than vision-based algorithms? A highly significant difference in training time and computational resource requirements is evident, with pose-based approaches significantly more efficient. However, it is important to note the overhead post-processing required to improve landmark accuracy and model performance.

3. Does applying transfer learning improve signer-independent SLR model performance? Transfer learning resulted in a 15.00% improvement in test accuracy on the AUTSL dataset and significantly reduced training time.

4. How does the number of video frames uniformly selected from each video clip affect model performance? The optimal number of frames was found to be 16 and increasing this did not significantly improve model performance and resulted in significantly longer training times. Fewer frames failed to capture spatiotemporal information from the videos.

6.2.2 Research Objectives

All seven objectives were successfully achieved during this study:

1. Isolated signer-independent SLR datasets were collected with a real-world complex signer and background setting. This was achieved through the isolated signer-independent AUTSL dataset.

2. A relatively inexpensive system was designed and created using real-world signer-independent data to perform isolated signer-independent SLR. RGB only input, as well as the computationally efficient unimodal pose-based and vision-based models on the AUTSL dataset, accomplished this.

3. Different feature extraction techniques were explored, comparing the performance of pose estimation, hand segmentation and full RGB input. The preliminary experiment compared the three feature extraction techniques on the LSA64 dataset, finding that pose landmark and full RGB were the best approaches for SLR.

4. The effectiveness of utilising frame interpolation and imputation on pose data was established. This was successfully achieved through the ablation experiment on the pose-based ST-GCN model, showing improved performance with frame interpolation and imputation on missing pose landmarks.

5. The relative effectiveness of vision-based approaches for signer-independent SLR was established. Various vision-based architectures were considered, with spatio-temporal convolutional networks and video vision transformers being the top-performing architectures.

6. The relative effectiveness of transfer learning on the vision-based models for signer-independent SLR was established. Transfer learning utilising the Kinetics-400 dataset resulted in a 15.00% accuracy improvement on the AUTSL dataset compared to training from scratch.

7. The relative effectiveness of pose-based and vision-based approaches for signer-independent SLR was established. The overall performance of pose-based and vision-based unimodal architectures for signer-independent SLR proved successful on the AUTSL dataset, achieving top performance compared to existing unimodal architectures utilising only RGB data.

The completed objectives yielded three significant contributions to the body of knowledge:

1. The implementation of video vision transformers for the purpose of SLR.

2. A thorough analysis of the effect of the number of frames selected for video tasks in deep learning.

3. A results contribution, including evaluating pose-based and vision-based architectures toward signer-independent SLR.

6.3 Future Work

Possible future work includes the following additional objectives:

1. The exploration of the models' performance on significantly larger datasets such as WLASL3000.
2. An analysis of the model performance when provided with different camera angles, i.e. face-on and side-on.
3. Effectively combine appropriate models to produce a multimodal state-of-the-art system that leverages the full potential of RGB input by combining feature extraction approaches to perform effective isolated signer-independent SLR.
4. Migrate the system from isolated SLR to continuous SLR to classify entire sign language sentences and conversations successfully.
5. Develop a mobile application for the system to be used in the field with readily available mobile input devices.

References

Adeyanju, I., Bello, O., and Adegboye, M. Machine learning methods for sign language recognition: A critical review and analysis. *Intelligent Systems with Applications*, 12:200056, 2021. ISSN 2667-3053. doi:https://doi.org/10.1016/j.iswa.2021.200056.

Ahmad, T., Jin, L., Zhang, X., Lai, S., Tang, G., and Lin, L. Graph convolutional neural network for human action recognition: A comprehensive survey. *IEEE Transactions on Artificial Intelligence*, 2(2):128–145, 2021. doi:10.1109/TAI.2021.3076974.

Al-qurishi, M., Khalid, T., and Souissi, R. Deep Learning for Sign Language Recognition : Current Techniques , Benchmarks , and Open Issues. *IEEE Access*, 9:126917–126951, 2021. doi:10.1109/ACCESS.2021.3110912.

Aloysius, N. and Geetha, M. Understanding vision-based continuous sign language recognition. *Multimedia Tools and Applications*, 79(31-32):22177–22209, 2020. ISSN 15737721. doi:10.1007/s11042-020-08961-z.

Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., and Grundmann, M. Blazepose: On-device real-time body pose tracking. *CoRR*, abs/2006.10204, 2020.

Bertasius, G., Wang, H., and Torresani, L. Is space-time attention all you need for video understanding? In *Proceedings of the International Conference on Machine Learning (ICML)*. July 2021.

Bhatt, D., Patel, C., Talsania, H., Patel, J., Vaghela, R., Pandya, S., Modi, K., and Ghayvat, H. Cnn variants for computer vision: History, architecture, application, challenges and future scope. *Electronics*, 10(20), 2021. ISSN 2079-9292. doi:10.3390/electronics10202470.

Boháček, M. and Hrúz, M. Sign pose-based transformer for word-level sign language recognition. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, pages 182–191. 2022. doi:10.1109/WACVW54805.2022.00024.

Camgoz, N. C., Koller, O., Hadfield, S., and Bowden, R. Sign language transformers: Joint end-to-end sign language recognition and translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10023–10033. 2020.

Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S., and Sheikh, Y. A. Openpose: Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

Carreira, J. and Zisserman, A. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308. 2017.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, Doha, Qatar, October 2014. doi:10.3115/v1/D14-1179.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*. 2014.

Cooper, H., Holt, B., and Bowden, R. Sign Language Recognition, pages 539–562. Springer London, London, 2011. ISBN 978-0-85729-997-0. doi:10.1007/978-0-85729-997-0_27.

Cunningham, P., Cord, M., and Delany, S. J. Supervised learning. In *Machine learning techniques for multimedia*, pages 21–49. Springer, 2008.

De Coster, M., Rushe, E., Holmes, R., Ventresque, A., and Dambre, J. Towards the extraction of robust sign embeddings for low resource sign language recognition. *arXiv preprint arXiv:2306.17558*, 2023.

De Coster, M., Van Herreweghe, M., and Dambre, J. Sign language recognition with transformer networks. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6018–6024. European Language Resources Association, Marseille, France, May 2020. ISBN 979-10-95546-34-4.

De Coster, M., Van Herreweghe, M., and Dambre, J. Isolated sign recognition from rgb video using pose flow and self-attention. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3436–3445. 2021. doi:10.1109/CVPRW53098.2021.00383.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. IEEE, 2009.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Feng, M. and Meunier, J. Skeleton graph-neural-network-based human action recognition: A survey. *Sensors*, 22(6), 2022. ISSN 1424-8220. doi:10.3390/s22062091.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-Gonzalez, P., and Garcia-Rodriguez, J. A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, 70:41–65, 2018. ISSN 1568-4946. doi:https://doi.org/10.1016/j.asoc.2018.05.018.

Géron, A. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media, Inc., 2019.

Ghosh, D. K., Chakrabarty, A., Moon, H., and Piran, M. J. A spatio-temporal graph convolutional network model for internet of medical things (iomt). *Sensors*, 22(21), 2022. ISSN 1424-8220. doi:10.3390/s22218438.

Graves, A. Long Short-Term Memory, pages 37–45. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-24797-2. doi:10.1007/978-3-642-24797-2_4.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., and Chen, T. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018. ISSN 0031-3203. doi:https://doi.org/10.1016/j.patcog.2017.10.013.

Hara, K., Kataoka, H., and Satoh, Y. Learning spatio-temporal features with 3D residual networks for action recognition. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 3154–3160. 2017. doi:10.1109/ICCVW.2017.373.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. 2016. doi:10.1109/CVPR.2016.90.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Hirafuji Neiva, D. and Zanchettin, C. Gesture recognition: A review focusing on sign language in a mobile context. *Expert Systems with Applications*, 103:159–183, 2018. ISSN 0957-4174. doi:https://doi.org/10.1016/j.eswa.2018.01.051.

Hochreiter, S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi:10.1162/neco.1997.9.8.1735.

Hossin, M. and Sulaiman, M. N. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1, 2015.

Hrúz, M., Gruber, I., Kanis, J., Boháček, M., Hlaváč, M., and Krňoul, Z. One model is not enough: Ensembles for isolated sign language recognition. *Sensors*, 22(13), 2022. ISSN 1424-8220. doi:10.3390/s22135043.

Huang, J., Zhou, W., Li, H., and Li, W. Sign language recognition using 3D convolutional neural networks. In *2015 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. 2015. doi:10.1109/ICME.2015.7177428.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 448–456. PMLR, Lille, France, 07–09 Jul 2015.

Jiang, S., Sun, B., Wang, L., Bai, Y., Li, K., and Fu, Y. Skeleton aware multi-modal sign language recognition. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 3408–3418, 2021. ISSN 21607516. doi:10.1109/CVPRW53098.2021.00380.

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M., and Zisserman, A. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., and Shah, M. Transformers in vision: A survey. *ACM Computing Surveys*, 54(10s), 2022. ISSN 0360-0300. doi:10.1145/3505244.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Konstantinidis, D., Dimitropoulos, K., and Daras, P. Sign language recognition based on hand and body skeletal data. In *2018-3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)*, pages 1–4. IEEE, 2018.

Kozlov, A., Andronov, V., and Gritsenko, Y. Lightweight network architecture for real-time action recognition. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, SAC '20, pages 2074–2080. Association for Computing Machinery, New York, NY, USA, 2020. ISBN 9781450368667. doi:10.1145/3341105.3373906.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

LeCun, Y., Bottou, L., Orr, G. B., and Müller, K. R. Efficient BackProp, pages 9–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. ISBN 978-3-540-49430-0. doi:10.1007/3-540-49430-8_2.

Li, D., Opazo, C. R., Yu, X., and Li, H. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. *Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020*, pages 1448–1458, 2020. doi:10.1109/WACV45572.2020.9093512.

Li, R., Liu, Z., and Tan, J. A survey on 3D hand pose estimation: Cameras, methods, and datasets. *Pattern Recognition*, 93:251–272, 2019. ISSN 0031-3203. doi:https://doi.org/10.1016/j.patcog.2019.04.026.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002. 2021. doi:10.1109/ICCV48922.2021.00986.

Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., and Hu, H. Video swin transformer. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3192–3201. 2022. doi:10.1109/CVPR52688.2022.00320.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*. 2019.

Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C., Yong, M. G., Lee, J., Chang, W., Hua, W., Georg, M., and Grundmann, M. Mediapipe: A framework for building perception pipelines. *CoRR*, abs/1906.08172, 2019.

Marais, M., Brown, D., Connan, J., and Boby, A. An evaluation of hand-based algorithms for sign language recognition. In *2022 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)*, pages 1–6. 2022. doi:10.1109/icABCD54961.2022.9856310.

Massone, M. I. and Curiel, M. Sign order in argentine sign language. *Sign Language Studies*, 5(1):63–93, 2004.

Mayberry, R. I. and Squires, B. Sign language acquisition. *Encyclopedia of language and linguistics*, 11:739–43, 2006.

Mor, B., Garhwal, S., and Kumar, A. A systematic review of hidden markov models and their applications. *Archives of Computational Methods in Engineering*, 28:1429–1448, 2021. doi:10.1007/s11831-020-09422-4.

Moryossef, A., Tsochantaridis, I., Dinn, J., Camgoz, N. C., Bowden, R., Jiang, T., Rios, A., Muller, M., and Ebling, S. Evaluating the immediate applicability of pose estimation for sign language recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3434–3440. 2021.

Munea, T. L., Jembre, Y. Z., Weldegebriel, H. T., Chen, L., Huang, C., and Yang, C. The progress of human pose estimation: a survey and taxonomy of models applied in 2D human pose estimation. *IEEE Access*, 8:133330–133348, 2020.

Ouassil, M.-A., Cherradi, B., Hamida, S., Errami, M., EL Gannour, O., and Raihani, A. A fake news detection system based on combination of word embedded techniques and hybrid deep learning model. *International Journal of Advanced Computer Science and Applications*, 13(10), 2022.

Peng, W., Shi, J., Varanka, T., and Zhao, G. Rethinking the st-gcns for 3D skeleton-based human action recognition. *Neurocomputing*, 454:45–53, 2021. ISSN 0925-2312. doi:https://doi.org/10.1016/j.neucom.2021.05.004.

Pigou, L., Dieleman, S., Kindermans, P.-J., and Schrauwen, B. Sign language recognition using convolutional neural networks. In *Computer Vision - ECCV 2014 Workshops*, pages 572–578. Springer International Publishing, Cham, 2015. ISBN 978-3-319-16178-5.

Prechelt, L. Automatic early stopping using cross validation: quantifying the criteria. *Neural Networks*, 11(4):761–767, 1998. ISSN 0893-6080. doi:https://doi.org/10.1016/S0893-6080(98)00010-0.

Prechelt, L. Early Stopping — But When?, pages 53–67. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-35289-8. doi:10.1007/978-3-642-35289-8_5.

Rastgoo, R., Kiani, K., and Escalera, S. Sign language recognition: A deep survey. *Expert Systems with Applications*, 164:113794, 2 2021a. ISSN 0957-4174. doi:10.1016/J.ESWA.2020.113794.

Rastgoo, R., Kiani, K., Escalera, S., and Sabokrou, M. Sign language production: A review. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3446–3456. 2021b. doi:10.1109/CVPRW53098.2021.00384.

Ronchetti, F., Quiroga, F., Estrebou, C., Lanzarini, L., and Rosete, A. Lsa64: A dataset of argentinian sign language. *XX II Congreso Argentino de Ciencias de la Computación (CACIC)*, 2016.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

Schuster, M. and Paliwal, K. K. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45, 1997.

Selva, J., Johansen, A. S., Escalera, S., Nasrollahi, K., Moeslund, T. B., and Clapés, A. Video transformers: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):12922–12943, 2023. doi:10.1109/TPAMI.2023.3243465.

Sengupta, A., Jin, F., Zhang, R., and Cao, S. mm-pose: Real-time human skeletal posture estimation using mmwave radars and cnns. *IEEE Sensors Journal*, 20(17):10032–10044, 2020. doi:10.1109/JSEN.2020.2991741.

Sharir, G., Noy, A., and Zelnik-Manor, L. An image is worth 16x16 words, what is a video worth? *arXiv preprint arXiv:2103.13915*, 2021.

Sharma, S., Gupta, R., and Kumar, A. Continuous sign language recognition using isolated signs data and deep transfer learning. *Journal of Ambient Intelligence and Humanized Computing*, 2021. ISSN 18685145. doi:10.1007/S12652-021-03418-Z.

Shi, L., Zhang, Y., Cheng, J., and Lu, H. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12018–12027. 2019. doi:10.1109/CVPR.2019.01230.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

Shoham, S. and Heber, M. Characteristics of a virtual community for individuals who are d/deaf and hard of hearing. *American Annals of the Deaf*, 157(3):251–263, 2012. ISSN 0002726X, 15430375.

Shorten, C. and Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019. doi:10.1186/s40537-019-0197-0.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Sincan, O. M., Junior, J., Jacques, C., Escalera, S., and Keles, H. Y. Chalearn lap large scale signer independent isolated sign language recognition challenge: Design,

results and future research. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3472–3481. 2021.

Sincan, O. M. and Keles, H. Y. Autsl: A large scale multi-modal turkish sign language dataset and baseline methods. *IEEE Access*, 8:181340–181355, 2020. doi: 10.1109/ACCESS.2020.3028072.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014. ISSN 1532-4435.

Stokoe, W. C. Sign language structure. *Annual Review of Anthropology*, 9(1):365–390, 1980. doi:10.1146/annurev.an.09.100180.002053.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9. 2015. doi:10.1109/CVPR.2015.7298594.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826. 2016. doi:10.1109/CVPR.2016.308.

Tachau, F. Language and politics: Turkish language reform. *The Review of Politics*, 26(2):191–204, 1964. ISSN 00346705, 17486858.

Tong, Z., Song, Y., Wang, J., and Wang, L. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems*. 2022.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., and Paluri, M. A closer look at spatiotemporal convolutions for action recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6450–6459. 2018. doi: 10.1109/CVPR.2018.00675.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Von Agris, U., Schneider, D., Zieren, J., and Kraiss, K.-F. Rapid signer adaptation for isolated sign language recognition. In *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, pages 159–159. 2006. doi: 10.1109/CVPRW.2006.165.

Von Agris, U., Zieren, J., Canzler, U., Bauer, B., and Kraiss, K.-F. Recent developments in visual sign language recognition. *Universal Access in the Information Society*, 6:323–362, 2008.

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., and Val Gool, L. Temporal segment networks: Towards good practices for deep action recognition. In *The European Conference on Computer Vision (ECCV)*. 2016.

Wiesler, S. and Ney, H. A convergence analysis of log-linear training. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6861–6871. PMLR, 09–15 Jun 2019.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021. doi:10.1109/TNNLS.2020.2978386.

Yan, S., Xiong, Y., and Lin, D. Spatial temporal graph convolutional networks for skeleton-based action recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. doi:10.1609/aaai.v32i1.12328.

Yim, J., Joo, D., Bae, J., and Kim, J. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *2017 IEEE Conference*

on Computer Vision and Pattern Recognition (CVPR), pages 7130–7138. 2017. doi: 10.1109/CVPR.2017.754.

Ying, X. An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168(2):022022, feb 2019. doi:10.1088/1742-6596/1168/2/022022.

Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C., and Grundmann, M. Mediapipe hands: On-device real-time hand tracking. *CoRR*, abs/2006.10214, 2020.

A

Code Listings

This appendix contains code listings relating to the architecture and system implementation described in Section 3.2

A.1 ST-GCN

```
class ST_GCN_block(nn.Module):
    def __init__(self, in_channels, out_channels, A, temporal_kernel_size, cuda_=False, stride=1, residual=True):
        super(ST_GCN_block, self).__init__()

        self.gcn = GraphConvolution(in_channels, out_channels, A, cuda_)
        self.tcn = TemporalConvolution(out_channels, out_channels, stride=stride, kernel_size=temporal_kernel_size)
        self.relu = nn.ReLU()
        if not residual:
            self.residual = lambda x: 0
        elif (in_channels == out_channels) and (stride == 1):
            self.residual = lambda x: x
        else:
            self.residual = TemporalConvolution(in_channels, out_channels, kernel_size=1, stride=stride)

    def forward(self, x):
        x = self.tcn(self.gcn(x)) + self.residual(x)
        return self.relu(x)
```

Listing A.1: ST-GCN block.

A.2 ResNet50-LSTM

```
class Resnet50_LSTM(nn.Module):

def __init__(self, Resnet50, LSTM):
    super(Resnet50_LSTM, self).__init__()
    self.Resnet50 = Resnet50
    self.LSTM = LSTM

def forward(self, x):
    """
    x (tensor): shape [batch_size, num_frames, channels, height, width]
    """
    x = self.Resnet50(x)
    x = self.LSTM(x)
    return x
```

Listing A.2: ResNet50-LSTM model with the sequential feeding of the ResNet50 model output features to the LSTM mode.

A.3 InceptionV3-LSTM

```
class InceptionV3LSTM(nn.Module):
def __init__(self, InceptionV3, LSTM):
    super(InceptionV3LSTM, self).__init__()
    self.InceptionV3 = InceptionV3
    self.LSTM = LSTM

def forward(self, x):
    """
    x (tensor): shape [batch_size, num_frames, channels, height, width]
    """
    x = self.InceptionV3(x)
    x = self.LSTM(x)
    return x
```

Listing A.3: InceptionV3-LSTM model with the sequential feeding of the InceptionV3 model output features to the LSTM model.

A.4 R(2+1)D-18

```
class BasicBlock(nn.Module):
    def __init__(self, in_planes, planes, stride=1):
        super(BasicBlock, self).__init__()
        self.conv1 = nn.Conv3d(in_planes, planes, kernel_size=(1, 3, 3), stride=(1, stride, stride), padding=(0, 1, 1), bias=False)
        self.bn1 = nn.BatchNorm3d(planes)
        self.relu = nn.ReLU(inplace=True)
        self.conv2 = nn.Conv3d(planes, planes, kernel_size=(3, 1, 1), stride=(stride, 1, 1), padding=(1, 0, 0), bias=False)
        self.bn2 = nn.BatchNorm3d(planes)

        self.shortcut = nn.Sequential()
        if stride != 1 or in_planes != planes:
            self.shortcut = nn.Sequential(
                nn.Conv3d(in_planes, planes, kernel_size=(1, 1, 1), stride=(1, stride, stride), bias=False),
                nn.BatchNorm3d(planes)
            )

    def forward(self, x):
        out = self.conv1(x)
        out = self.bn1(out)
        out = self.relu(out)
        out = self.conv2(out)
        out = self.bn2(out)
        out += self.shortcut(x)
        out = self.relu(out)
        return out
```

Listing A.4: R(2+1)D BasicBlock.

A.5 Early Stopping

```
    def __call__(self, val_loss, model, path):

        score = -val_loss

        if self.best_score is None:
            self.best_score = score
            self.save_checkpoint(val_loss, model, path)
        elif score < self.best_score + self.delta:
            self.counter += 1
            self.trace_func(f'EarlyStopping counter: {self.counter} out of {self.patience}')

            if self.counter >= self.patience:
                self.early_stop = True

            if self.counter <= 5:
                self.save_5_consecutive_checkpoints(val_loss, model, path)
        else:
            self.best_score = score
            self.save_checkpoint(val_loss, model, path)
            self.counter = 0
])
```

Listing A.5: PyTorch early stopping callback.

A.6 Model Training

```
    early_stopping = EarlyStopping(patience=patience, verbose=True)

    for epoch in range(epochs):
        model.train()
        for batch_idx, (data, label) in enumerate(train_dataloader):
            # extract inputs and labels
            data = data.to(device)
            label = label.to(device)

            # forward pass
            output = model(data.float())
            loss = criterion(output, label)
            loss = loss / accum_iter
            loss.backward()

            # weights update
            if ((batch_idx + 1)%accum_iter == 0) or (batch_idx + 1 == len(train_dataloader)):
                optimiser.step()
                optimiser.zero_grad()
```

Listing A.6: Model training pipeline.

A.7 Model Evaluation

```
    y_true = []
    y_pred = []
    model.eval()
    with torch.no_grad():
        for data, labels in test_dataloader:
            data = data.to(device)
            labels = labels.to(device)

            # calculate outputs by running video through the model
            outputs = model(data.float())
            _, predicted = torch.max(outputs.data, 1)

            # append ground truth outputs and predicted values to respective lists
            y_true.extend(labels.cpu().numpy())
            y_pred.extend(predicted.cpu().numpy())

    accuracy_score = metrics.accuracy_score(y_true, y_pred)
    f_score = metrics.f1_score(y_true, y_pred, average="weighted")
```

Listing A.7: Model testing and accuracy reporting.

B

Additional Results

B.1 Experiment 2.1 – ST-GCN Classification Report LSA64

Table B.1: Experiment 2.1 – ST-GCN classification report for LSA64 dataset.

Class	Precision	Recall	F1	Class	Precision	Recall	F1
Opaque	100.0	100.0	100.0	Hungry	100.0	100.0	100.0
Red	90.91	100.0	95.24	Map	100.0	100.0	100.0
Green	100.0	90.00	94.74	Coin	100.0	100.0	100.0
Yellow	100.0	100.0	100.0	Music	100.0	100.0	100.0
Bright	100.0	30.00	46.15	Ship	100.0	100.0	100.0
Light blue	100.0	100.0	100.0	None	90.91	100.0	95.24
Colors	83.33	100.0	90.91	Name	90.91	100.0	95.24
Pink	88.89	80.00	84.21	Patience	100.0	100.0	100.0
Women	100.0	100.0	100.0	Perfume	90.91	100.0	95.24
Enemy	90.91	100.0	95.24	Deaf	75.00	90.00	81.82
Son	100.0	90.00	94.74	Trap	90.91	100.0	95.24
Man	47.62	100.0	64.52	Rice	100.0	100.0	100.0
Away	100.0	100.0	100.0	Barbecue	100.0	90.00	94.74
Drawer	83.33	50.00	62.50	Candy	90.91	100.0	95.24
Born	83.33	50.00	62.50	Chewing gum	100.0	100.0	100.0
Learn	100.0	100.0	100.0	Spaghetti	100.0	100.0	100.0
Call	90.91	100.0	95.24	Yogurt	71.43	50.00	58.82
Skimmer	57.14	80.00	66.67	Accept	100.0	100.0	100.0
Bitter	90.91	100.0	95.24	Thanks	100.0	80.00	88.89
Sweet milk	100.0	100.0	100.0	Shut down	100.0	100.0	100.0
Milk	100.0	100.0	100.0	Appear	100.0	80.00	88.89

Class	Precision	Recall	F1	Class	Precision	Recall	F1
Water	100.0	80.00	88.89	To land	90.91	100.0	95.24
Food	90.91	100.0	95.24	Catch	100.0	70.00	82.35
Argentina	100.0	100.0	100.0	Help	100.0	100.0	100.0
Uruguay	100.0	100.0	100.0	Dance	100.0	100.0	100.0
Country	100.0	80.00	88.89	Bathe	100.0	90.00	94.74
Last name	83.33	100.0	90.91	Buy	56.25	90.00	69.23
Where	100.0	70.00	82.35	Copy	100.0	100.0	100.0
Mock	100.0	100.0	100.0	Run	90.91	100.0	95.24
Birthday	100.0	70.00	82.35	Realize	100.0	50.00	66.67
Breakfast	88.89	80.00	84.21	Give	100.0	100.0	100.0
Photo	83.33	100.0	90.91	Find	66.67	100.0	80.00

B.2 Experiment 2.2 – ST-GCN Classification Report AUTSL

Table B.2: Experiment 2.2 – ST-GCN classification report for AUTSL dataset.

Class	Precision	Recall	F1	Class	Precision	Recall	F1
Sister	100.0	93.75	96.77	Hurry	93.33	87.50	90.32
Hungry	93.34	100.0	100.0	EnjoyYourMeal	93.35	100.0	100.0
Brother	93.36	94.12	91.43	Tree	93.37	88.24	90.91
Heavy	93.38	58.82	64.52	Cry	93.39	76.47	86.67
Family	93.40	58.82	68.97	Wise	93.41	94.12	94.12
Unwise	93.42	100.0	85.00	Kin	93.43	100.0	100.0
Shopping	93.44	93.33	96.55	Key	93.45	93.75	96.77
Mother	93.46	100.0	100.0	Friend	93.47	100.0	96.77
Atatürk	93.48	92.86	72.22	Shoe	93.49	81.25	65.00
Mirror	93.50	94.12	94.12	Same	93.51	94.12	96.97
Father	93.52	93.75	93.75	Garden	93.53	100.0	100.0

Class	**Precision**	**Recall**	**F1**	**Class**	**Precision**	**Recall**	**F1**
Look	93.54	94.12	88.89	Honey	93.55	93.75	93.75
Glass	93.56	75.00	80.00	Flag	93.57	87.50	87.50
Feast	93.58	100.0	100.0	Baby	93.59	82.35	90.32
Single	93.60	100.0	100.0	Wait	93.61	100.0	100.0
I	93.62	58.82	74.07	Petrol	93.63	100.0	94.44
Together	93.64	88.24	88.24	Inform	93.65	94.12	96.97
We	93.66	100.0	94.44	Work	93.67	100.0	96.97
Wednesday	93.68	100.0	82.93	Fork	93.69	88.24	88.24
Tea	93.70	94.12	88.89	Teapot	93.71	81.25	86.67
Hammer	93.72	100.0	82.93	Ugly	93.73	94.12	96.97
Child	93.74	94.12	86.49	Soup	93.75	100.0	72.34
Friday	93.76	100.0	100.0	Saturday	93.77	94.12	96.97
Wallet	93.78	82.35	90.32	Minute	93.79	78.57	84.62
Grandfather	93.80	94.12	94.12	Change	93.81	100.0	94.44
Topple	93.82	100.0	100.0	Government	93.83	37.50	54.55
Doctor	93.84	80.00	84.21	Full	93.85	82.35	77.78
Wedding	93.86	93.75	88.24	Yesterday	93.87	70.59	72.73
Enemy	93.88	100.0	94.12	Wall	93.89	81.25	89.66
Pharmacy	93.90	100.0	87.18	Glove	93.91	94.12	82.05
Labor	93.92	100.0	96.55	Retired	93.93	88.24	93.75
Male	93.94	82.35	73.68	Meal	93.95	88.24	93.75
House	93.96	94.12	96.97	Yes	93.97	100.0	93.75
Married	93.98	93.75	93.75	Memorize	93.99	80.00	82.76
Elephant	94.00	93.75	96.77	Photograph	94.01	100.0	100.0
Football	94.02	94.12	91.43	Past	94.03	100.0	96.97
Get Well	94.04	100.0	96.97	Bring	94.05	94.12	86.49
Lake	94.06	68.75	78.57	Shirt	94.07	93.75	93.75
See	94.08	94.12	96.97	Show	94.09	100.0	100.0
Laugh	94.10	94.12	91.43	Lightweight	94.11	72.73	64.00
Right	94.12	100.0	94.44	Carpet	94.13	88.24	93.75
Ill	94.14	82.35	84.85	Hospital	94.15	100.0	100.0

Class	**Precision**	**Recall**	**F1**	**Class**	**Precision**	**Recall**	**F1**
Fault	94.16	94.12	96.97	Towel	94.17	100.0	100.0
No	94.18	94.12	84.21	Congratulations	94.19	100.0	82.35
Animal	94.20	100.0	100.0	Gift	94.21	88.24	93.75
Halal	94.22	94.12	94.12	Always	94.23	88.24	90.91
Never	94.24	70.59	82.76	Goodbye	94.25	100.0	97.14
Drink	94.26	100.0	82.93	Needle	94.27	80.00	85.71
Medicine	94.28	58.82	74.07	Not Interested	94.29	52.94	64.29
Light	94.30	100.0	82.05	Push	94.31	100.0	91.89
Good	94.32	81.25	89.66	Escape	94.33	82.35	90.32
Breakfast	94.34	100.0	100.0	Pencil	94.35	93.75	85.71
Radiator	94.36	88.24	88.24	Door	94.37	100.0	100.0
Sibling	94.38	100.0	89.47	Crossroads	94.39	88.24	88.24
Accident	94.40	93.75	85.71	Belt	94.41	70.59	82.76
If Only	94.42	70.59	77.42	Who	94.43	76.47	76.47
Identity	94.44	100.0	87.18	Rent	94.45	100.0	100.0
Book	94.46	100.0	97.14	Mince	94.47	100.0	100.0
Female	94.48	100.0	94.44	Smell	94.49	92.86	96.30
Cologne	94.50	100.0	100.0	Coal	94.51	100.0	100.0
Dog	94.52	100.0	89.47	Bridge	94.53	100.0	97.14
Bad	94.54	82.35	82.35	Lap	94.55	94.12	86.49
Stain	94.56	88.24	93.75	Salary	94.57	94.12	69.57
Scissors	94.58	93.33	96.55	Tongs	94.59	93.75	93.75
God Preserve	94.60	88.24	88.24	Angel	94.61	82.35	90.32
Be Pleased	94.62	94.12	96.97	Napkin	94.63	57.14	66.67
Stairs	94.64	100.0	90.91	Guest	94.65	94.12	78.05
Manager	94.66	100.0	94.44	Tap	94.67	94.12	91.43
How	94.68	87.50	90.32	Why	94.69	76.47	78.79
Where	94.70	94.12	91.43	Grandmother	94.71	100.0	97.14
Oven	94.72	76.47	86.67	Room	94.73	100.0	100.0
Wood	94.74	76.47	86.67	Teacher	94.75	100.0	97.14
School	94.76	41.18	46.67	Olympiad	94.77	100.0	97.14

Class	**Precision**	**Recall**	**F1**	**Class**	**Precision**	**Recall**	**F1**
Nope	94.78	76.47	83.87	Allright	94.79	94.12	82.05
They	94.80	64.71	66.67	Forest	94.81	100.0	94.44
Fasting	94.82	37.50	50.00	Apologize	94.83	100.0	91.89
Cotton	94.84	100.0	100.0	Trousers	94.85	100.0	97.14
Money	94.86	94.12	94.12	Pastrami	94.87	94.12	91.43
Potato	94.88	94.12	86.49	Sunday	94.89	88.24	93.75
Monday	94.90	100.0	100.0	Window	94.91	94.12	96.97
Thursday	94.92	94.12	91.43	Picnic	94.93	76.47	72.22
Police	94.94	100.0	91.89	Psychology	94.95	52.94	62.07
Request	94.96	100.0	97.14	Hour	94.97	82.35	87.50
Soap	94.98	70.59	66.67	Sauce	94.99	100.0	100.0
Tuesday	95.00	100.0	100.0	Champion	95.01	88.24	81.08
Hat	95.02	88.24	88.24	War	95.03	64.71	75.86
Sugar	95.04	63.64	63.64	Hi	95.05	64.71	78.57
Umbrella	95.06	76.47	83.87	You	95.07	100.0	89.47
Bill	95.08	100.0	96.77	Free	95.09	100.0	100.0
Voice	95.10	94.12	86.49	Love	95.11	100.0	100.0
Evil	95.12	100.0	100.0	Border	95.13	100.0	97.14
You (Plural)	95.14	75.00	77.42	Say	95.15	75.00	85.71
Promise	95.16	88.24	90.91	Milk	95.17	100.0	97.14
Okay	95.18	93.75	90.91	Comb	95.19	58.82	74.07
Date	95.20	88.24	90.91	Holiday	95.21	94.12	96.97
Sweet	95.22	88.24	88.24	Ceiling	95.23	100.0	100.0
Danger	95.24	64.71	75.86	Telephone	95.25	88.24	88.24
Scales	95.26	88.24	78.95	Tailor	95.27	33.33	47.62
Thanks	95.28	88.24	93.75	Screwdriver	95.29	100.0	100.0
Turkey	95.30	93.75	90.91	Orange	95.31	88.24	93.75
Toilet	95.32	82.35	90.32	Flour	95.33	88.24	93.75
Far	95.34	100.0	100.0	Sad	95.35	100.0	97.14
Existing	95.36	94.12	96.97	Tax	95.37	52.94	66.67
Near	95.38	71.43	80.00	Alone	95.39	64.71	68.75

Class	Precision	Recall	F1	Class	Precision	Recall	F1
Wrong	95.40	94.12	91.43	Do	95.41	94.12	88.89
Band-Aid	95.42	64.71	78.57	Help	95.43	100.0	100.0
Tomorrow	95.44	82.35	80.00	Forbidden	95.45	82.35	90.32
Pillow	95.46	94.12	96.97	Bed	95.47	100.0	97.14
Slow	95.48	52.94	64.29	Eat	95.49	68.75	62.86
Cook	95.50	100.0	94.44	Star	95.51	100.0	91.89
Absent	95.52	100.0	100.0	Road	95.53	88.24	93.75
Tired	95.54	100.0	100.0	Egg	95.55	88.24	90.91
Time	95.56	94.12	88.89	Difficult	95.57	82.35	87.50

B.3 Experiment 3.1 – R(2+1)D-18 Classification Report LSA64

Table B.3: Experiment 3.1 – R(2+1)D-18 classification report for LSA64 dataset.

Class	Precision	Recall	F1	Class	Precision	Recall	F1
Opaque	100.00	100.00	100.00	Hungry	100.00	100.00	100.00
Red	100.00	100.00	100.00	Map	100.00	100.00	100.00
Green	100.00	100.00	100.00	Coin	100.00	100.00	100.00
Yellow	100.00	100.00	100.00	Music	100.00	100.00	100.00
Bright	100.00	100.00	100.00	Ship	100.00	100.00	100.00
Light blue	100.00	100.00	100.00	None	100.00	100.00	100.00
Colors	100.00	100.00	100.00	Name	100.00	100.00	100.00
Pink	100.00	100.00	100.00	Patience	100.00	100.00	100.00
Women	100.00	100.00	100.00	Perfume	100.00	100.00	100.00
Enemy	100.00	100.00	100.00	Deaf	90.91	100.00	95.24
Son	100.00	100.00	100.00	Trap	100.00	100.00	100.00
Man	100.00	100.00	100.00	Rice	100.00	100.00	100.00
Away	100.00	100.00	100.00	Barbecue	100.00	100.00	100.00

Class	Precision	Recall	F1	Class	Precision	Recall	F1
Drawer	100.00	100.00	100.00	Candy	100.00	100.00	100.00
Born	100.00	100.00	100.00	Chewing gum	100.00	90.00	94.74
Learn	100.00	100.00	100.00	Spaghetti	100.00	100.00	100.00
Call	90.91	100.00	95.24	Yogurt	100.00	100.00	100.00
Skimmer	100.00	100.00	100.00	Accept	100.00	100.00	100.00
Bitter	100.00	100.00	100.00	Thanks	100.00	100.00	100.00
Sweet milk	100.00	100.00	100.00	Shut down	100.00	100.00	100.00
Milk	100.00	100.00	100.00	Appear	100.00	100.00	100.00
Water	100.00	90.00	94.74	To land	100.00	100.00	100.00
Food	100.00	100.00	100.00	Catch	100.00	100.00	100.00
Argentina	100.00	100.00	100.00	Help	100.00	100.00	100.00
Urugauy	100.00	100.00	100.00	Dance	100.00	100.00	100.00
Country	100.00	100.00	100.00	Bathe	100.00	100.00	100.00
Last name	100.00	100.00	100.00	Buy	100.00	100.00	100.00
Where	100.00	100.00	100.00	Copy	100.00	100.00	100.00
Mock	100.00	100.00	100.00	Run	100.00	100.00	100.00
Birthday	100.00	100.00	100.00	Realize	100.00	100.00	100.00
Breakfast	100.00	100.00	100.00	Give	100.00	100.00	100.00
Photo	100.00	100.00	100.00	Find	100.00	100.00	100.00

B.4 Experiment 3.2 – R(2+1)D-18 Classification Report AUTSL

Table B.4: Experiment 3.2 – R(2+1)D-18 classification report for AUTSL dataset.

Class	Precision	Recall	F1	Class	Precision	Recall	F1
Sister	100.0	81.25	89.66	Hurry	100.0	100.0	100.0
Hungry	89.47	100.0	94.44	EnjoyYourMeal	100.0	94.12	96.97
Brother	100.0	100.0	100.0	Tree	80.00	94.12	86.49

Class	**Precision**	**Recall**	**F1**	**Class**	**Precision**	**Recall**	**F1**
Heavy	66.67	47.06	55.17	Cry	92.31	70.59	80.00
Family	92.31	70.59	80.00	Wise	100.0	94.12	96.97
Unwise	80.95	100.0	89.47	Kin	94.12	100.0	96.97
Shopping	92.86	86.67	89.66	Key	78.95	93.75	85.71
Mother	94.44	100.0	97.14	Friend	93.75	100.0	96.77
Atatürk	73.68	100.0	84.85	Shoe	76.47	81.25	78.79
Mirror	93.75	88.24	90.91	Same	100.0	76.47	86.67
Father	100.0	93.75	96.77	Garden	88.89	94.12	91.43
Look	75.00	88.24	81.08	Honey	100.0	75.00	85.71
Glass	76.47	81.25	78.79	Flag	77.78	87.50	82.35
Feast	94.12	94.12	94.12	Baby	100.0	70.59	82.76
Single	100.0	100.0	100.0	Wait	100.0	100.0	100.0
I	92.31	70.59	80.00	Petrol	100.0	100.0	100.0
Together	93.33	82.35	87.50	Inform	100.0	88.24	93.75
We	93.75	88.24	90.91	Work	88.89	100.0	94.12
Wednesday	94.44	100.0	97.14	Fork	94.12	94.12	94.12
Tea	100.0	94.12	96.97	Teapot	100.0	93.75	96.77
Hammer	100.0	100.0	100.0	Ugly	94.12	94.12	94.12
Child	73.91	100.0	85.00	Soup	85.00	100.0	91.89
Friday	100.0	94.12	96.97	Saturday	100.0	94.12	96.97
Wallet	100.0	94.12	96.97	Minute	55.56	71.43	62.50
Grandfather	100.0	100.0	100.0	Change	85.00	100.0	91.89
Topple	100.0	100.0	100.0	Government	81.25	81.25	81.25
Doctor	63.64	70.00	66.67	Full	100.0	100.0	100.0
Wedding	85.71	75.00	80.00	Yesterday	100.0	70.59	82.76
Enemy	81.25	81.25	81.25	Wall	92.86	81.25	86.67
Pharmacy	100.0	100.0	100.0	Glove	100.0	88.24	93.75
Labor	100.0	100.0	100.0	Retired	100.0	82.35	90.32
Male	85.00	100.0	91.89	Meal	94.44	100.0	97.14
House	100.0	100.0	100.0	Yes	88.24	100.0	93.75
Married	100.0	93.75	96.77	Memorize	86.67	86.67	86.67

Class	**Precision**	**Recall**	**F1**	**Class**	**Precision**	**Recall**	**F1**
Elephant	82.35	87.50	84.85	Photograph	93.75	100.0	96.77
Football	100.0	100.0	100.0	Past	75.00	93.75	83.33
Get Well	93.33	87.50	90.32	Bring	73.33	64.71	68.75
Lake	94.12	100.0	96.97	Shirt	100.0	100.0	100.0
See	100.0	88.24	93.75	Show	100.0	100.0	100.0
Laugh	93.33	82.35	87.50	Lightweight	40.74	100.0	57.89
Right	80.95	100.0	89.47	Carpet	93.75	88.24	90.91
Ill	94.44	100.0	97.14	Hospital	100.0	100.0	100.0
Fault	80.95	100.0	89.47	Towel	100.0	100.0	100.0
No	100.0	94.12	96.97	Congratulations	82.35	100.0	90.32
Animal	94.12	100.0	96.97	Gift	83.33	88.24	85.71
Halal	100.0	64.71	78.57	Always	88.89	94.12	91.43
Never	84.21	94.12	88.89	Goodbye	92.31	70.59	80.00
Drink	77.27	100.0	87.18	Needle	100.0	100.0	100.0
Medicine	100.0	88.24	93.75	Not Interested	73.33	64.71	68.75
Light	66.67	100.0	80.00	Push	76.19	94.12	84.21
Good	100.0	87.50	93.33	Escape	100.0	88.24	93.75
Breakfast	100.0	82.35	90.32	Pencil	86.67	81.25	83.87
Radiator	100.0	76.47	86.67	Door	84.21	94.12	88.89
Sibling	85.00	100.0	91.89	Crossroads	88.89	94.12	91.43
Accident	100.0	81.25	89.66	Belt	91.67	64.71	75.86
If Only	93.75	88.24	90.91	Who	78.57	64.71	70.97
Identity	100.0	94.12	96.97	Rent	93.33	82.35	87.50
Book	84.21	94.12	88.89	Mince	100.0	100.0	100.0
Female	77.27	100.0	87.18	Smell	76.47	92.86	83.87
Cologne	100.0	100.0	100.0	Coal	100.0	100.0	100.0
Dog	100.0	100.0	100.0	Bridge	89.47	100.0	94.44
Bad	82.35	82.35	82.35	Lap	89.47	100.0	94.44
Stain	100.0	88.24	93.75	Salary	70.83	100.0	82.93
Scissors	100.0	86.67	92.86	Tongs	70.00	87.50	77.78
God Preserve	82.35	82.35	82.35	Angel	100.0	88.24	93.75

Class	**Precision**	**Recall**	**F1**	**Class**	**Precision**	**Recall**	**F1**
Be Pleased	100.0	94.12	96.97	Napkin	76.92	71.43	74.07
Stairs	80.00	80.00	80.00	Guest	100.0	100.0	100.0
Manager	94.44	100.0	97.14	Tap	93.33	82.35	87.50
How	100.0	75.00	85.71	Why	100.0	70.59	82.76
Where	88.89	94.12	91.43	Grandmother	94.44	100.0	97.14
Oven	100.0	94.12	96.97	Room	100.0	100.0	100.0
Wood	93.75	88.24	90.91	Teacher	100.0	94.12	96.97
School	68.42	76.47	72.22	Olympiad	94.44	100.0	97.14
Nope	88.89	94.12	91.43	Allright	87.50	82.35	84.85
They	57.69	88.24	69.77	Forest	89.47	100.0	94.44
Fasting	88.89	50.00	64.00	Apologize	100.0	100.0	100.0
Cotton	94.44	100.0	97.14	Trousers	84.21	94.12	88.89
Money	100.0	82.35	90.32	Pastrami	100.0	94.12	96.97
Potato	100.0	100.0	100.0	Sunday	100.0	100.0	100.0
Monday	93.75	88.24	90.91	Window	100.0	100.0	100.0
Thursday	94.44	100.0	97.14	Picnic	89.47	100.0	94.44
Police	69.57	94.12	80.00	Psychology	89.47	100.0	94.44
Request	100.0	82.35	90.32	Hour	100.0	58.82	74.07
Soap	87.50	82.35	84.85	Sauce	93.75	93.75	93.75
Tuesday	100.0	94.12	96.97	Champion	89.47	100.0	94.44
Hat	94.44	100.0	97.14	War	100.0	100.0	100.0
Sugar	66.67	72.73	69.57	Hi	88.89	94.12	91.43
Umbrella	100.0	88.24	93.75	You	88.24	88.24	88.24
Bill	100.0	73.33	84.62	Free	89.47	100.0	94.44
Voice	93.75	88.24	90.91	Love	94.12	94.12	94.12
Evil	100.0	100.0	100.0	Border	89.47	100.0	94.44
You (Plural)	100.0	50.00	66.67	Say	80.00	100.0	88.89
Promise	100.0	100.0	100.0	Milk	82.35	82.35	82.35
Okay	92.31	75.00	82.76	Comb	100.0	94.12	96.97
Date	100.0	94.12	96.97	Holiday	87.50	82.35	84.85
Sweet	87.50	82.35	84.85	Ceiling	94.44	100.0	97.14

Class	Precision	Recall	F1	Class	Precision	Recall	F1
Danger	93.75	88.24	90.91	Telephone	100.0	100.0	100.0
Scales	93.33	82.35	87.50	Tailor	92.31	80.00	85.71
Thanks	100.0	88.24	93.75	Screwdriver	100.0	100.0	100.0
Turkey	84.21	100.0	91.43	Orange	89.47	100.0	94.44
Toilet	89.47	100.0	94.44	Flour	100.0	94.12	96.97
Far	94.12	100.0	96.97	Sad	88.89	94.12	91.43
Existing	94.12	94.12	94.12	Tax	100.0	76.47	86.67
Near	100.0	92.86	96.30	Alone	100.0	82.35	90.32
Wrong	100.0	100.0	100.0	Do	78.95	88.24	83.33
Band-Aid	94.12	94.12	94.12	Help	100.0	100.0	100.0
Tomorrow	81.25	76.47	78.79	Forbidden	100.0	58.82	74.07
Pillow	88.24	88.24	88.24	Bed	100.0	100.0	100.0
Slow	88.89	94.12	91.43	Eat	100.0	100.0	100.0
Cook	100.0	100.0	100.0	Star	100.0	100.0	100.0
Absent	80.95	100.0	89.47	Road	87.50	82.35	84.85
Tired	100.0	100.0	100.0	Egg	100.0	100.0	100.0
Time	80.00	94.12	86.49	Difficult	100.0	100.0	100.0

B.5 Experiment 2 and 3 – Confusion Matrices

The confusion matrix results for Experiments 2 and 3 are available on the following Google Drive for the interested reader. Due to their large size, they could not be depicted within the thesis.

C

Datasets

This appendix contains figures relating to the breakdown of the datasets utilised for signer-independent SLR.

C.1 AUTSL Dataset

C.1.1 AUTSL Backgrounds

The AUTSL dataset contains 20 different challenging background variations. Some of the backgrounds we also recorded with the camera field-of-view altered by adding or removing some objects to or from the background scene. Figure C.1 visualises examples of the different backgrounds from the AUTSL dataset. As depicted in the illustration, backgrounds present numerous challenges. In outdoor recordings, dynamic backgrounds may include moving elements such as trees or people passing behind the signer. The videos encompass a range of lighting conditions, spanning from sunlight to artificial light. Consequently, the video frames exhibit variations in illumination, along with areas that may be shadowed or brightly lit.

C.1.2 AUTSL Sign Classes

Table C.1 provides the distribution of the number of samples of each sign within the train split of the AUTSL dataset. The number of samples for each sign ranges between 90 and 127, with an average of 124 samples. The signs *Sugar* and *Okay* are the two most underrepresented classes, with 90 and 95 samples each. Furthermore, the number of hands used to perform each sign is included.

Figure C.1: Examples of the 20 different background variations from AUTSL.

Table C.1: AUTSL sign classes with the number of hands used to perform each sign and the number of training samples of each sign.

English Sign	**Hand**	**Train Samples**	**English Sign**	**Hand**	**Train Samples**
Sister	Single	126	Rent	Both	125
Hurry	Both	125	Book	Both	126
Hungry	Single	123	Mince	Both	119
E.Y.M.	Single	123	Female	Single	125
Brother	Single	125	Smell	Single	127
Tree	Both	125	Cologne	Both	121
Heavy	Both	125	Coal	Both	124
Cry	Single	122	Dog	Both	127
Family	Both	125	Bridge	Both	127
Wise	Single	122	Bad	Single	126
Unwise	Single	121	Lap	Both	127
Kin	Both	126	Stain	Both	126
Shopping	Both	127	Salary	Both	124
Key	Both	124	Scissors	Single	125
Mother	Single	119	Tongs	Single	127
Friend	Both	121	God Preserve	Both	127
Ataturk	Single	108	Angel	Both	126
Shoe	Both	123	Be Pleased	Both	125
Mirror	Single	120	Napkin	Single	125
Same	Single	120	Stairs	Single	127
Father	Single	125	Guest	Single	126
Garden	Both	125	Manager	Single	126
Look	Single	122	Tap	Single	127
Honey	Both	121	How	Single	126
Glass	Single	123	Why	Both	127
Flag	Both	126	Where	Both	127
Feast	Single	126	Grandmother	Single	127
Baby	Both	127	Oven	Both	127
Single	Single	126	Room	Both	126

English Sign	**Hand**	**Train Samples**	**English Sign**	**Hand**	**Train Samples**
Wait	Single	125	Wood	Both	124
I	Single	127	Teacher	Single	124
Petrol	Both	127	School	Single	126
Together	Both	126	Olympiad	Both	127
Inform	Both	127	Nope	Single	126
We	Both	118	Allright	Single	125
Work	Both	126	They	Single	123
Wednesday	Single	124	Forest	Both	127
Fork	Both	126	Fasting	Single	127
Tea	Both	126	Apologize	Both	127
Teapot	Both	127	Cotton	Both	126
Hammer	Both	127	Trousers	Both	126
Ugly	Single	126	Money	Single	126
Child	Single	125	Pastrami	Both	125
Soup	Single	127	Potato	Single	125
Friday	Single	126	Sunday	Both	125
Saturday	Both	126	Monday	Single	127
Wallet	Both	126	Window	Both	127
Minute	Both	124	Thursday	Single	126
Grandfather	Single	126	Picnic	Both	125
Change	Both	127	Police	Both	126
Topple	Both	125	Psychology	Single	127
Government	Single	121	Request	Single	127
Doctor	Both	112	Hour	Both	125
Full	Single	127	Soap	Both	127
Wedding	Both	120	Sauce	Both	127
Yesterday	Single	122	Tuesday	Both	127
Enemy	Single	123	Champion	Both	124
Wall	Both	112	Hat	Single	126
Pharmacy	Both	120	War	Both	126
Glove	Both	122	Sugar	Single	90

English Sign	Hand	Train Samples	English Sign	Hand	Train Samples
Labor	Single	123	Hi	Single	127
Retired	Single	122	Umbrella	Both	124
Male	Single	125	You	Single	123
Meal	Single	120	Bill	Both	120
House	Both	127	Free	Both	127
Yes	Single	125	Voice	Single	126
Married	Both	122	Love	Single	125
Memorize	Single	123	Evil	Both	127
Elephant	Both	120	Border	Both	124
Photograph	Both	124	You (Plural)	Both	121
Football	Both	121	Say	Single	126
Past	Single	124	Promise	Single	125
Get Well	Both	124	Milk	Single	127
Bring	Single	124	Okay	Both	95
Lake	Both	122	Comb	Single	126
Shirt	Both	122	Date	Both	127
See	Single	122	Holiday	Both	127
Show	Both	125	Sweet	Single	126
Laugh	Single	126	Ceiling	Both	126
Lightweight	Both	116	Danger	Both	127
Right	Single	122	Telephone	Single	127
Carpet	Both	127	Scales	Both	127
Ill	Single	127	Tailor	Both	126
Hospital	Both	126	Thanks	Both	126
Fault	Single	126	Screwdriver	Both	127
Towel	Both	127	Turkey	Single	127
No	Single	121	Orange	Both	126
Congratulations	Single	121	Toilet	Single	126
Animal	Both	126	Flour	Both	127
Gift	Single	123	Far	Both	126
Halal	Both	126	Sad	Single	127

English Sign	**Hand**	**Train Samples**	**English Sign**	**Hand**	**Train Samples**
Always	Both	124	Existing	Single	127
Never	Single	126	Tax	Both	124
Goodbye	Single	127	Near	Single	120
Drink	Single	122	Alone	Single	127
Needle	Single	118	Wrong	Single	125
Medicine	Single	123	Do	Both	126
Not Interested	Both	126	Band-Aid	Both	126
Light	Single	121	Help	Both	125
Push	Both	126	Tomorrow	Single	117
Good	Single	126	Forbidden	Both	123
Escape	Single	124	Pillow	Both	124
Breakfast	Both	125	Bed	Both	127
Pencil	Single	125	Slow	Both	126
Radiator	Both	127	Eat	Single	126
Door	Both	126	Cook	Both	123
Sibling	Both	126	Star	Both	127
Crossroads	Both	126	Absent	Single	126
Accident	Both	125	Road	Single	127
Belt	Both	127	Tired	Both	126
If Only	Single	127	Egg	Both	122
Who	Single	126	Time	Both	123
Identity	Both	126	Difficult	Both	120